Lecture 07

Cloud Computing

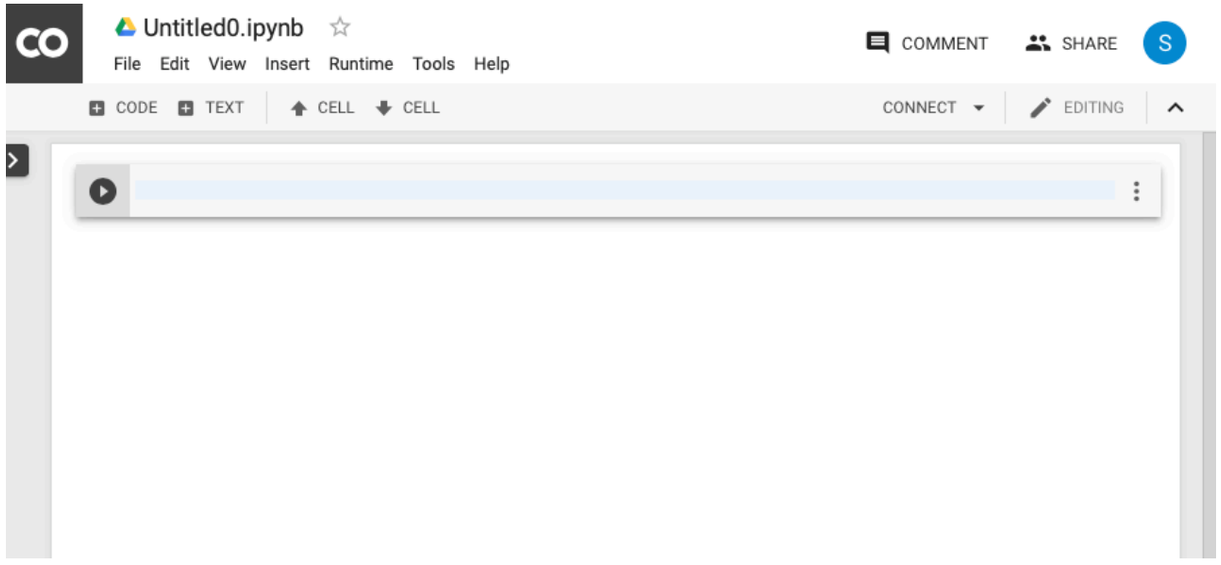
STAT 479: Deep Learning, Spring 2019

Sebastian Raschka

http://stat.wisc.edu/~sraschka/teaching/stat479-ss2019/

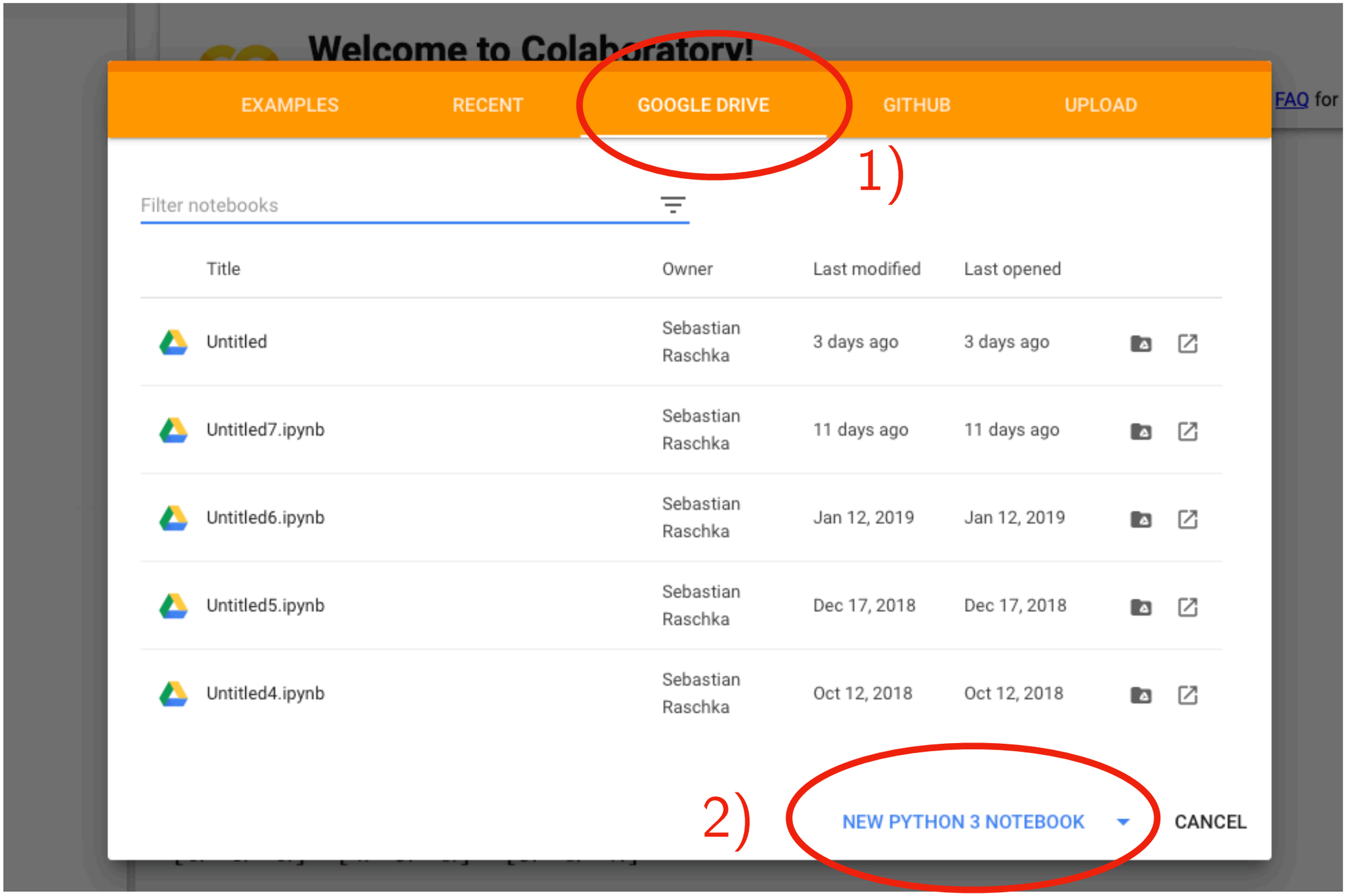
Option 1: Google Colab

https://colab.research.google.com



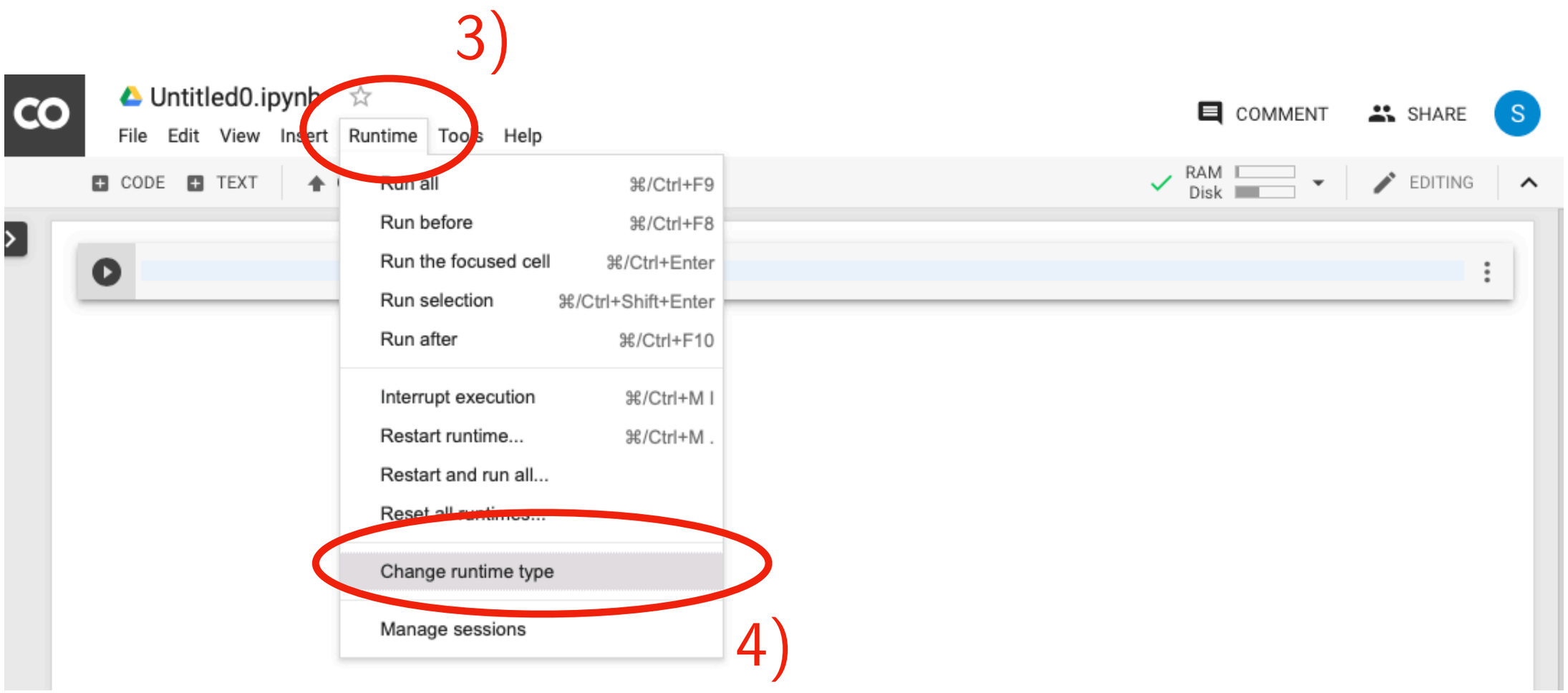
- Free Google-flavored Jupyter Notebooks in the Cloud
- For each notebook, they spin up a custom (Linux-based) computing instance
- Computations limited to ~12 h though; you won't lose your notebook, but computations will be interrupted
- Maybe useful for quick testing/experimenting/sharing (but maybe tedious as you need to reinstall packages each time)

Option 1: Google Colab



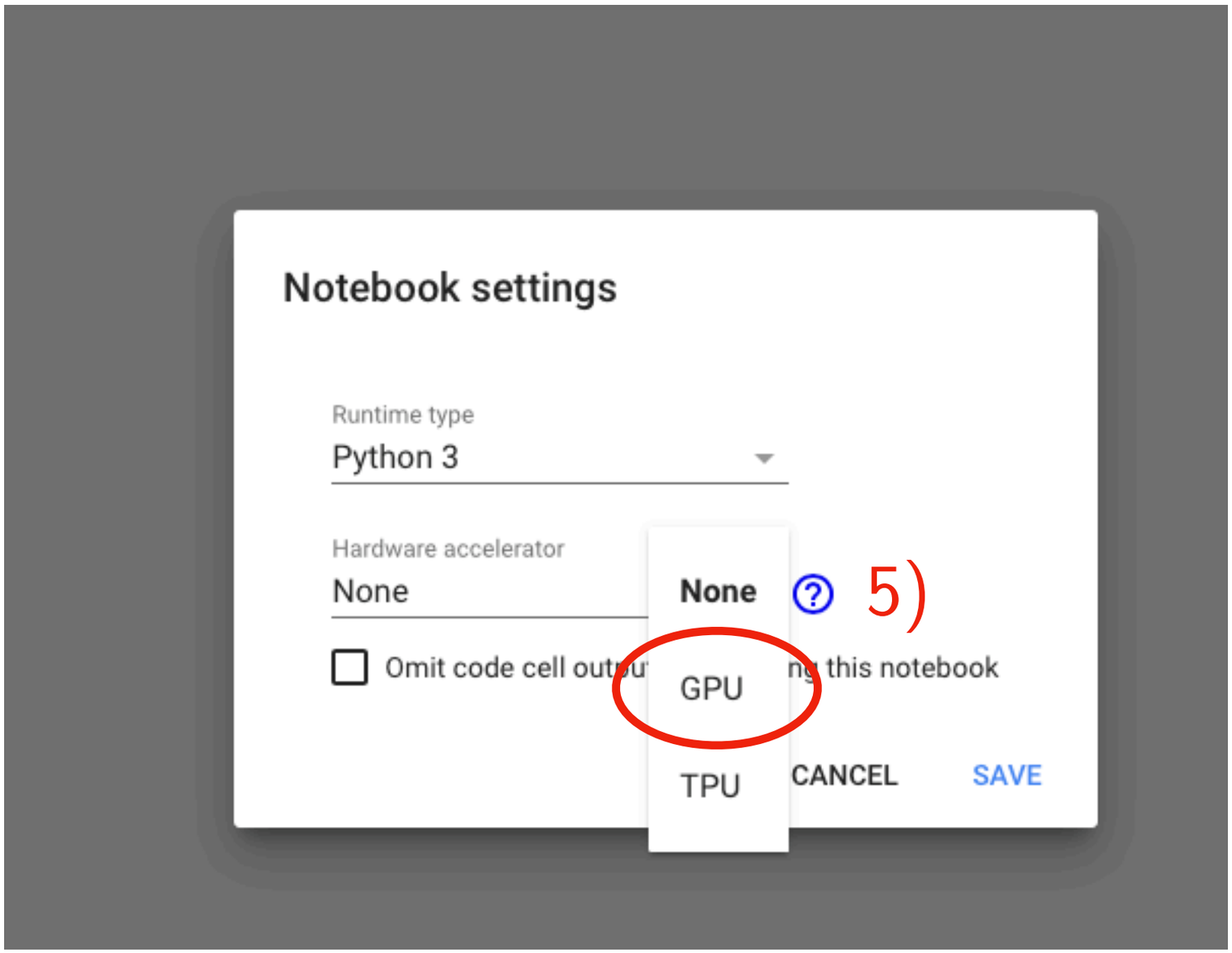
Menu appears if you visit https://colab.research.google.com

Option 1: Google Colab



Follow these steps for running code on GPU later (default is CPU)

Option 1: Google Colab



Follow these steps for running code on GPU later (default is CPU)

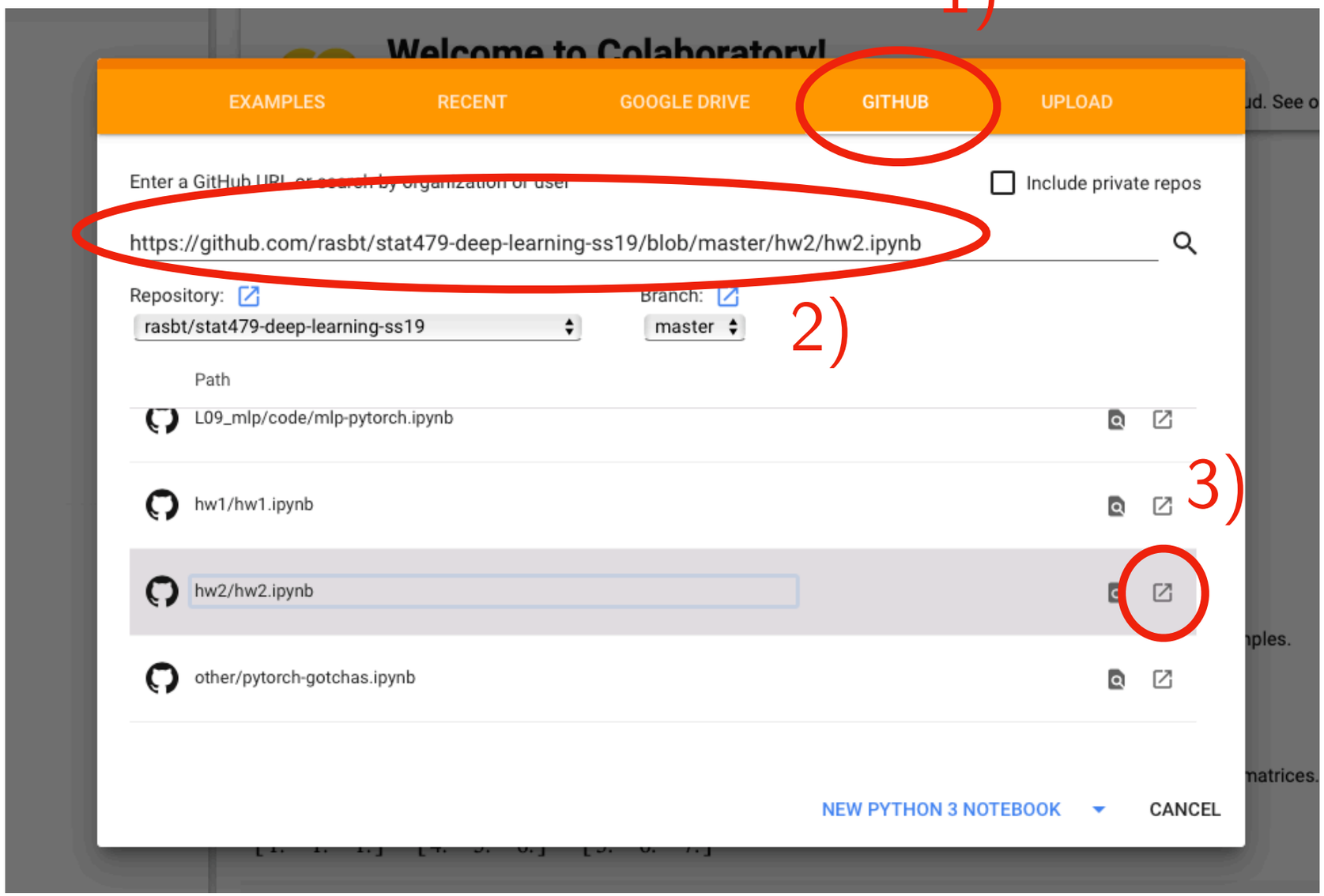
Option 1: Google Colab

- This is NEW! It appears that PyTorch is already pre-installed now (it wasn't always the case)

Option 1: Google Colab

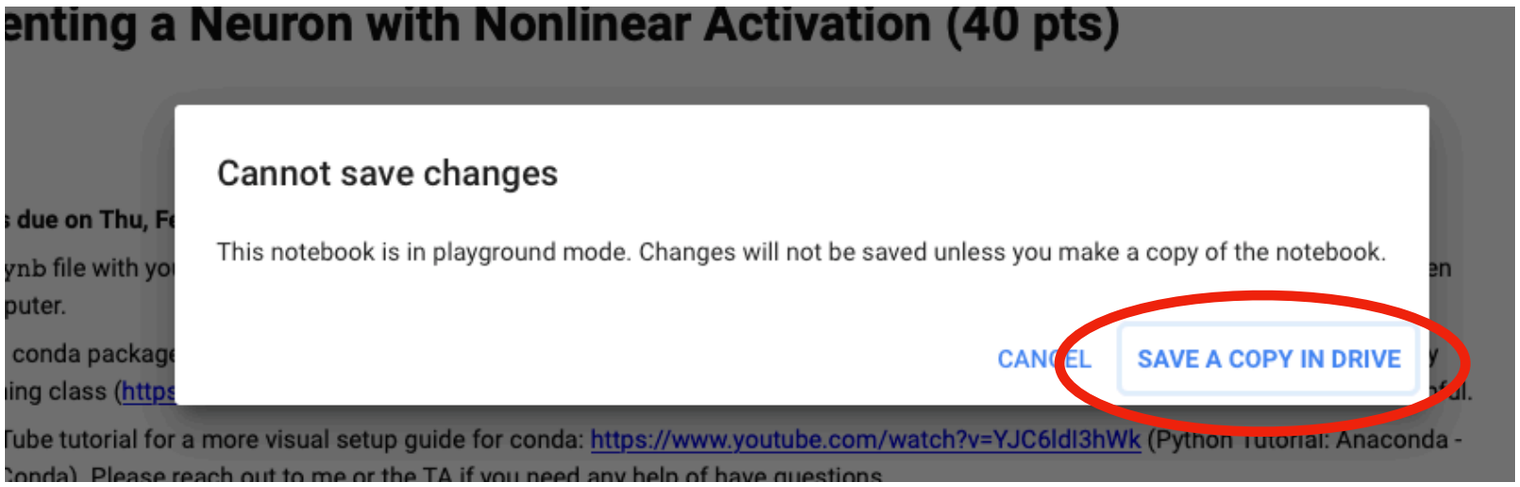
- In any case, if you'd like/need to install packages, you can do it as shown in the example above
- Note that in Jupyter Notebooks, the "!" indicates that what follows on that line is a "shell command" (you can think of a "shell" as the Linux & macOS command-line terminal, e.g,. a Bash Shell)

Option 1: Google Colab



- You can also upload Notebooks or directly paste GitHub links to notebooks

Option 1: Google Colab



4)

When you import a Notebook from a GitHub link, make sure to save it in your Google Drive if you plan to make edits, otherwise it will be gone later

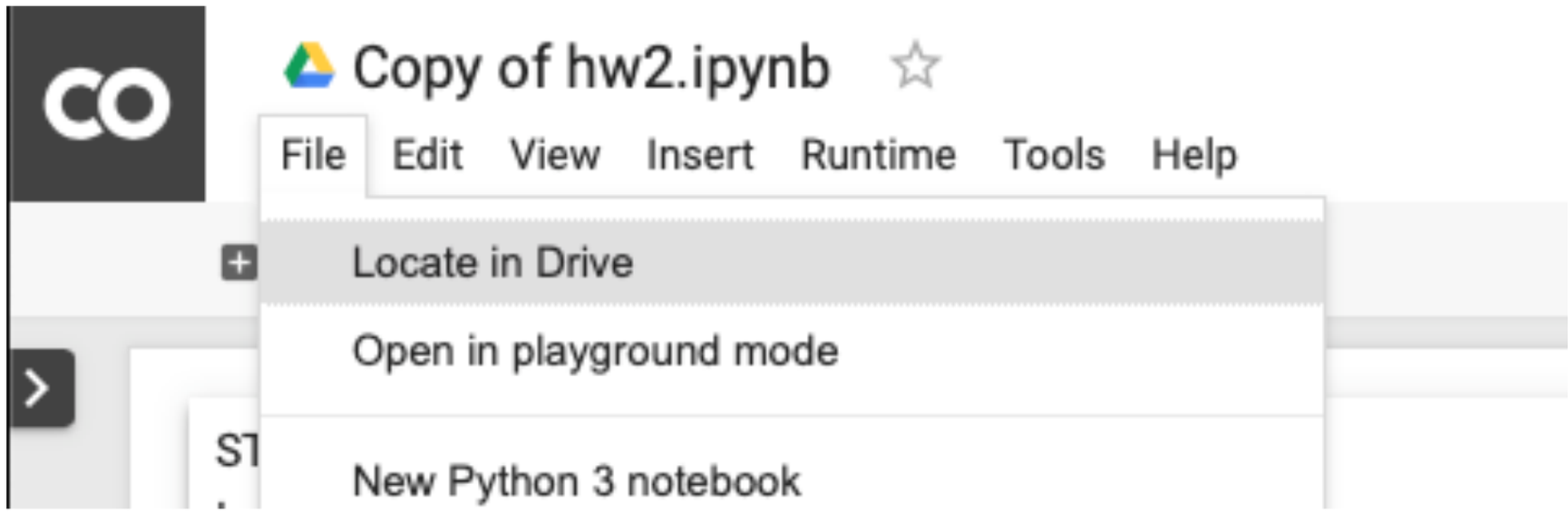
Option 1: Google Colab

```
/usr/local/lib/python3.6/dist-packages/pandas/io/parsers.py in __init__(self, f, engine, **kwds)
    816             self.options['has_index_names'] = kwds['has_index_names']
    817
--> 818         self._make_engine(self.engine)
    819
    820     def close(self):

/usr/local/lib/python3.6/dist-packages/pandas/io/parsers.py in _make_engine(self, engine)
   1047     def _make_engine(self, engine='c'):
   1048         if engine == 'c':
-> 1049             self._engine = CParserWrapper(self.f, **self.options)
   1050         else:
   1051             if engine == 'python':

/usr/local/lib/python3.6/dist-packages/pandas/io/parsers.py in __init__(self, src, **kwds)
   1693         kwds['allow_leading_cols'] = self.index_col is not False
   1694
-> 1695         self._reader = parsers.TextReader(src, **kwds)
   1696
   1697         # XXX

pandas/_libs/parsers.pyx in pandas._libs.parsers.TextReader.__cinit__()

pandas/_libs/parsers.pyx in pandas._libs.parsers.TextReader._setup_parser_source()

FileNotFoundError: File b'./datasets/iris.data' does not exist

SEARCH STACK OVERFLOW
```

If you'd run the HW2 notebook, you'd likely encounter this error.
This is because it can't find the dataset via the specified, relative link ...

Option 1: Google Colab

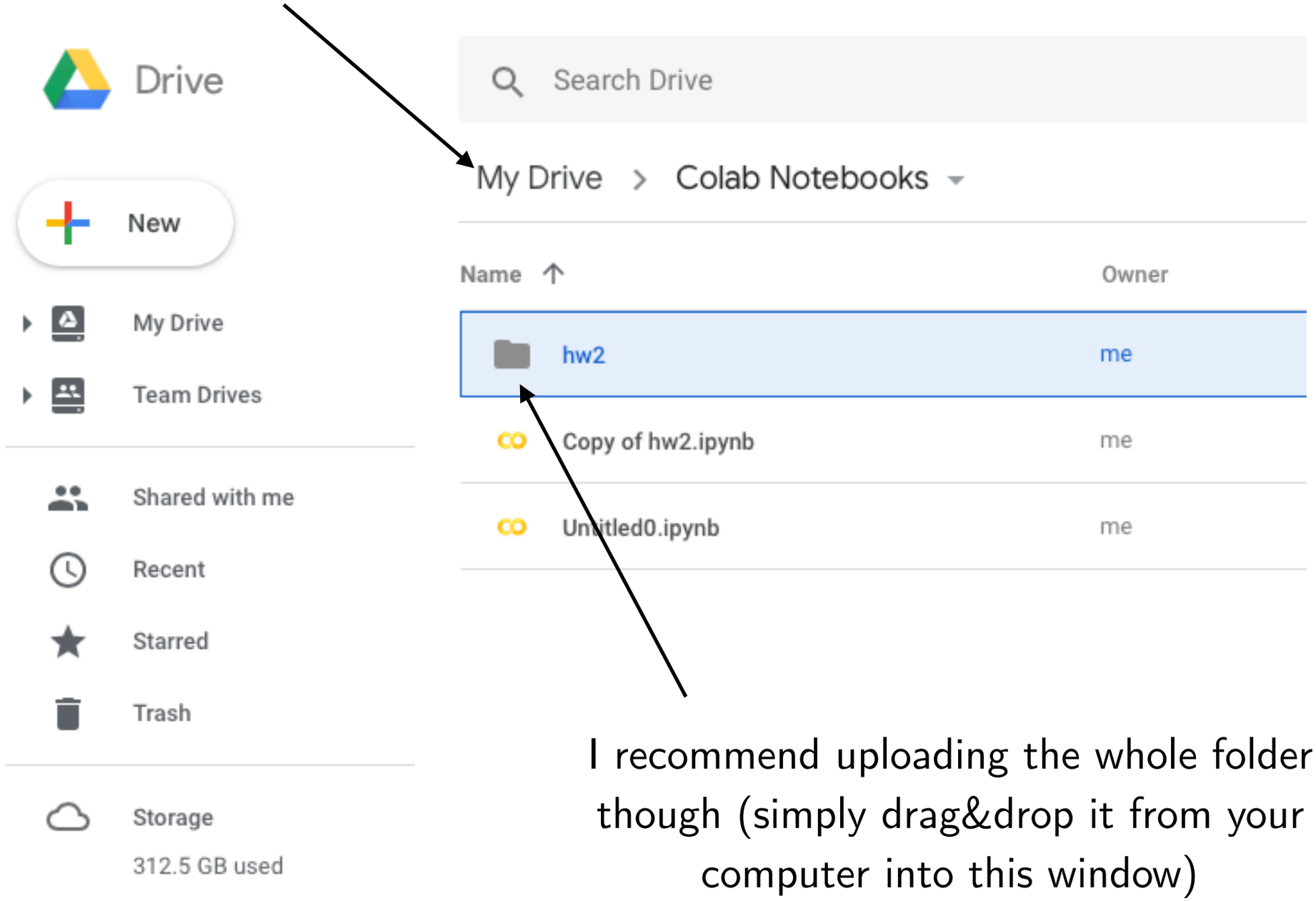
... you'd need to get the datafile into the same location as the notebook*.
First, locate the position of the saved Notebook in your Google Drive:



*technically, it is also possible to load CSV files via pandas.read_csv, but getting the dataset onto Google Drive may be generally useful, e.g., for working with more complex datasets later.

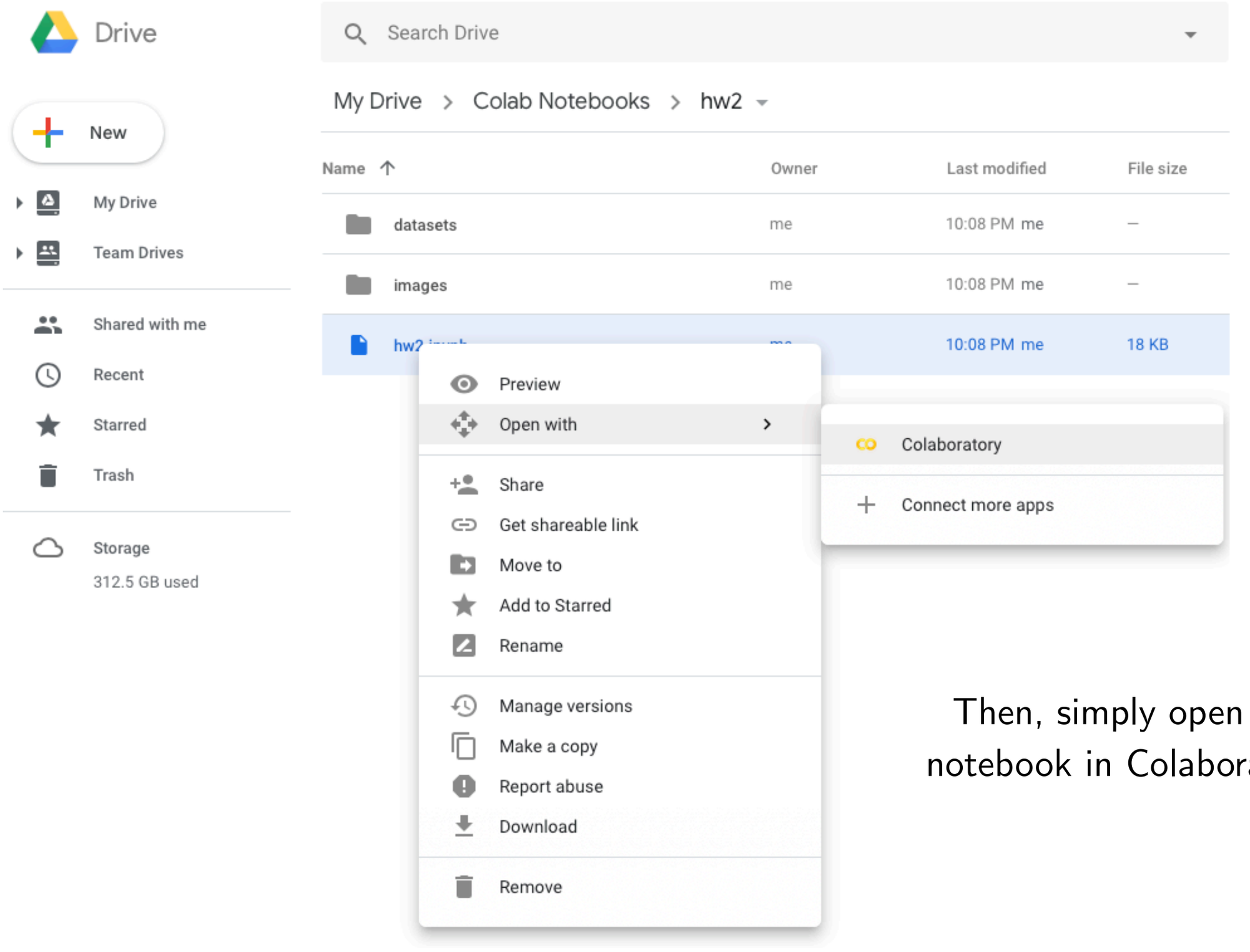
Option 1: Google Colab

Notebooks are usually in a directory called "My Drive/Colab Notebooks"



I recommend uploading the whole folder though (simply drag&drop it from your computer into this window)

Option 1: Google Colab



Then, simply open the notebook in Colaboratory.

Option 1: Google Colab

1) Unfortunately, there's some extra step required: mounting your Google Drive to the computer that now runs the Notebook. You need to execute the following code:

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318
Enter your authorization code:
```

Then, click on the link and enter it in the field above

2)

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Go to this URL in a browser: https://accounts.google.co
Enter your authorization code:
··········
```

3)

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Go to this URL in a browser: https://ac
Enter your authorization code:
··········
Mounted at /content/drive
```

Your Google Drive should now be finally mounted:

Option 1: Google Colab

Now, you simply need to provide the correct address to the dataset inside the Notebook and it should work:

```
[16] df = pd.read_csv('/content/drive/My Drive/Colab Notebooks/hw2/datasets/iris.data', index_col=None, header=None)
     df.columns = ['x1', 'x2', 'x3', 'x4', 'y']
     df = df.iloc[50:150]
     df['y'] = df['y'].apply(lambda x: 0 if x == 'Iris-versicolor' else 1)
     df.tail()
```

	x1	x2	x3	x4	y
145	6.7	3.0	5.2	2.3	1
146	6.3	2.5	5.0	1.9	1
147	6.5	3.0	5.2	2.0	1
148	6.2	3.4	5.4	2.3	1
149	5.9	3.0	5.1	1.8	1

Option 2: Google Cloud Instances

This is trickier and you don't have to use it for this class, but it's a useful skill and experience!

https://console.cloud.google.com/education

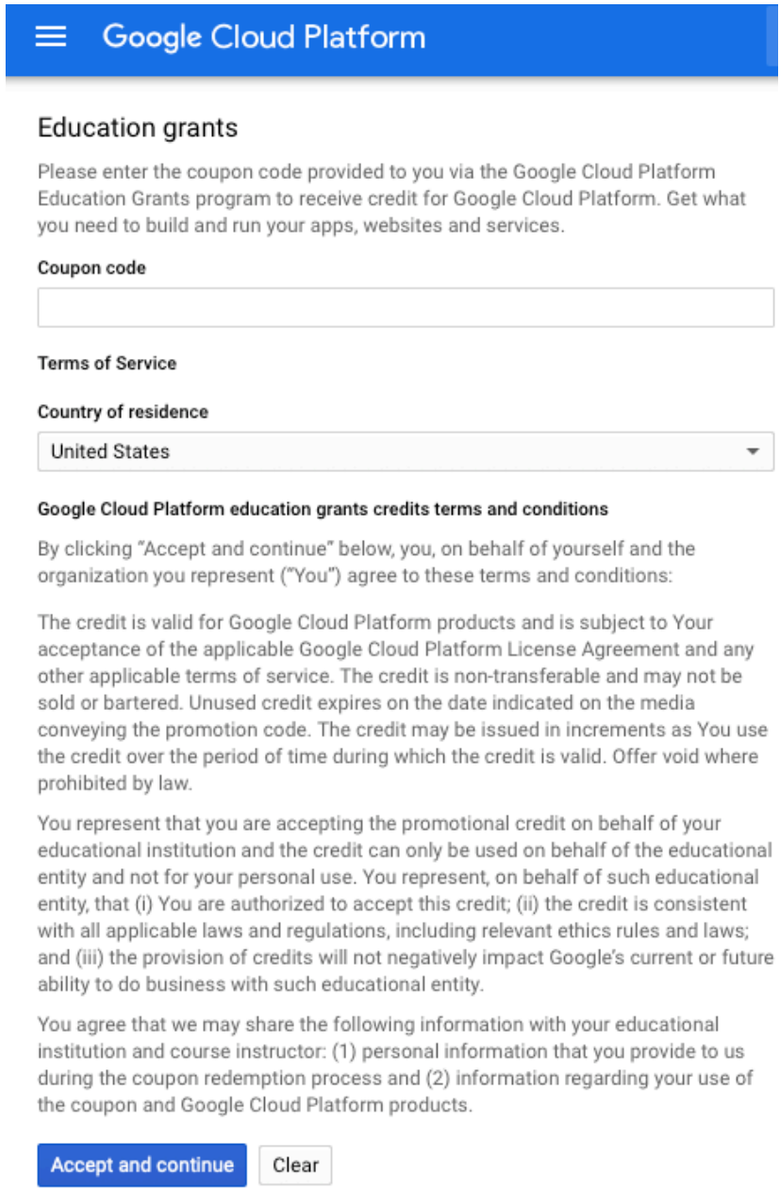
Will email a
$50 coupon code (per student) after class

Option 2: Google Cloud Instances

Go to the website

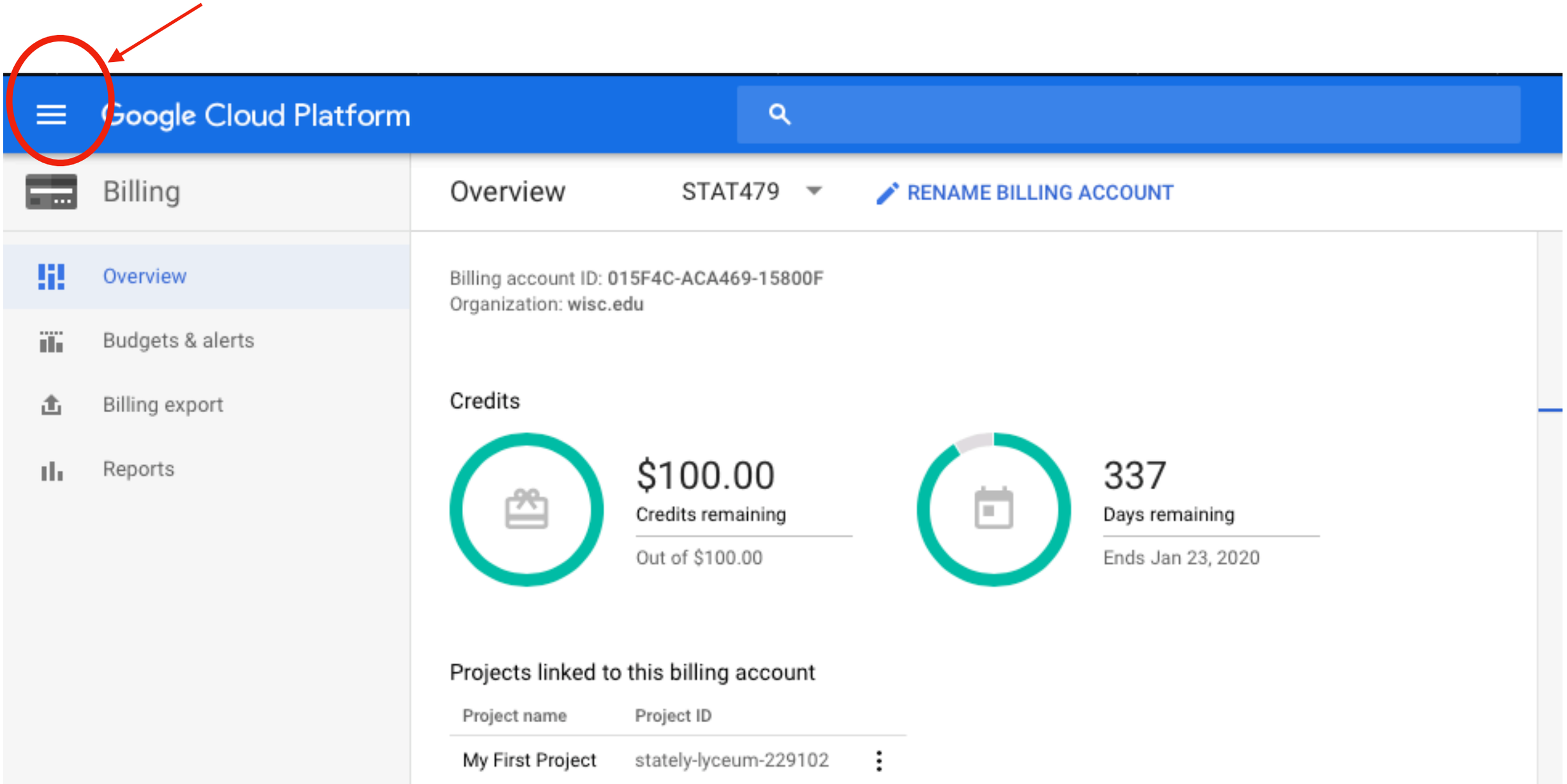
https://console.cloud.google.com/education

Read the terms, and accept if you agree
(you don't have to use this platform for this class!)



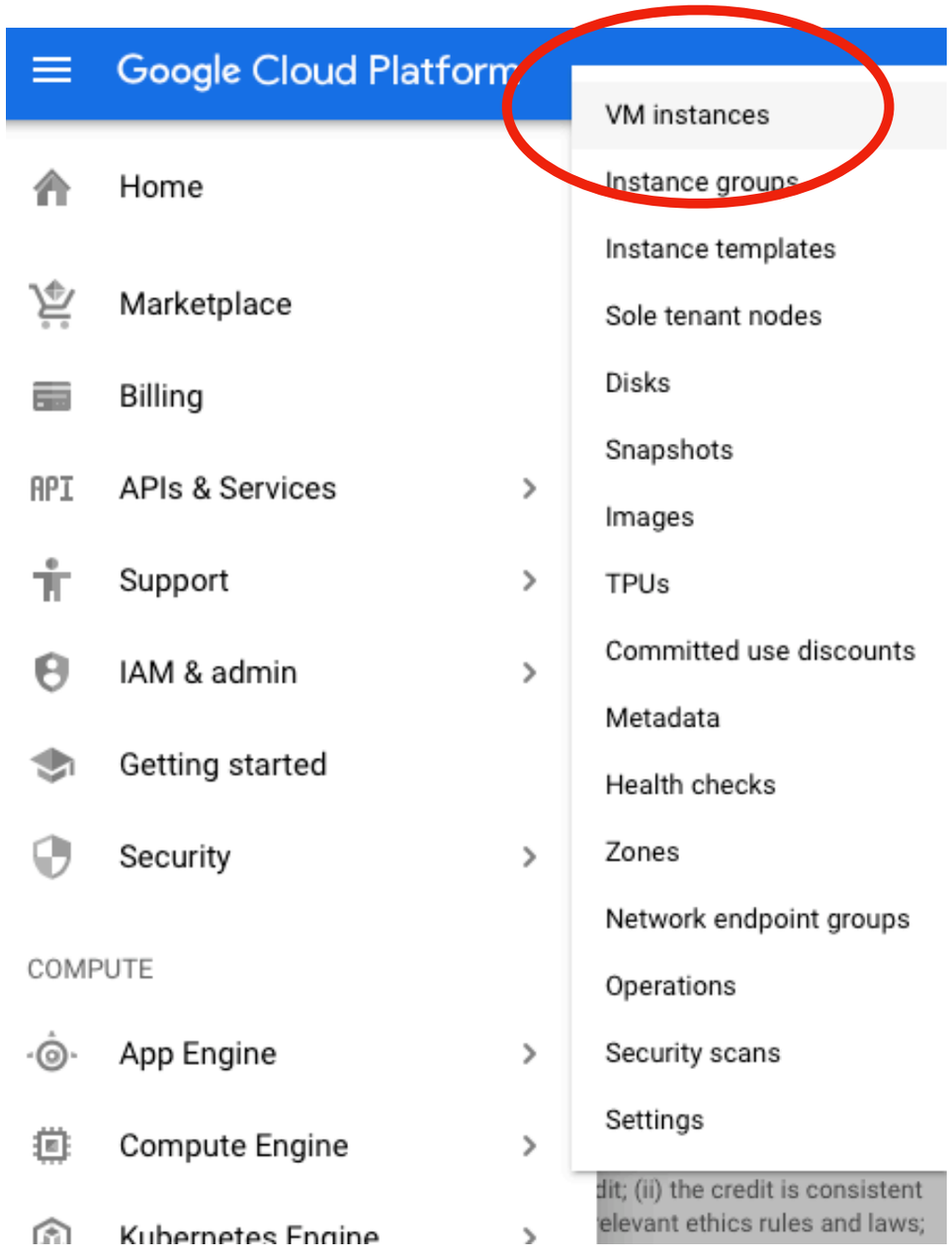
Option 2: Google Cloud Instances

Check your credits periodically, via the billings menu that can be accessed from the main menu



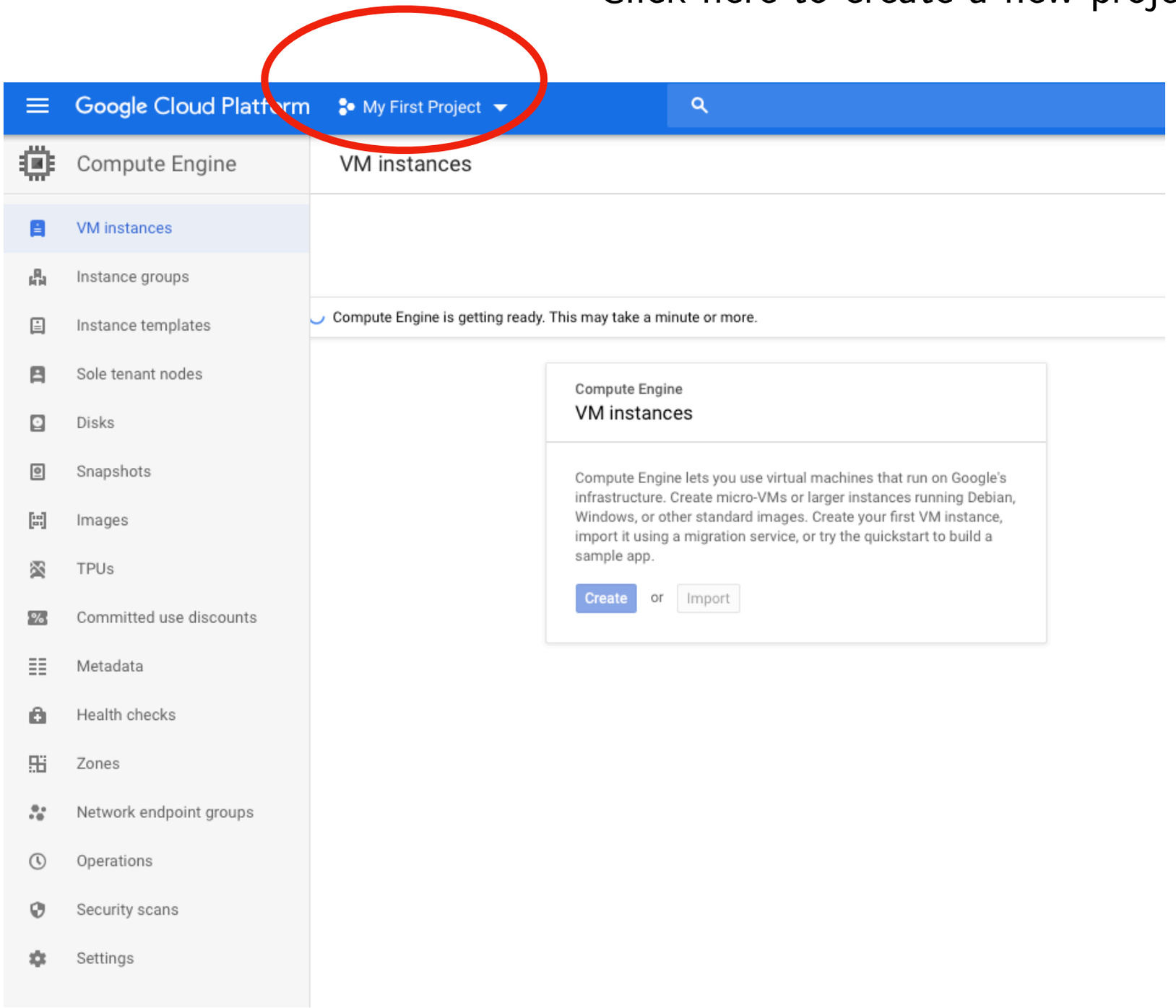
Option 2: Google Cloud Instances

To create a new computing instance, click on VM instances

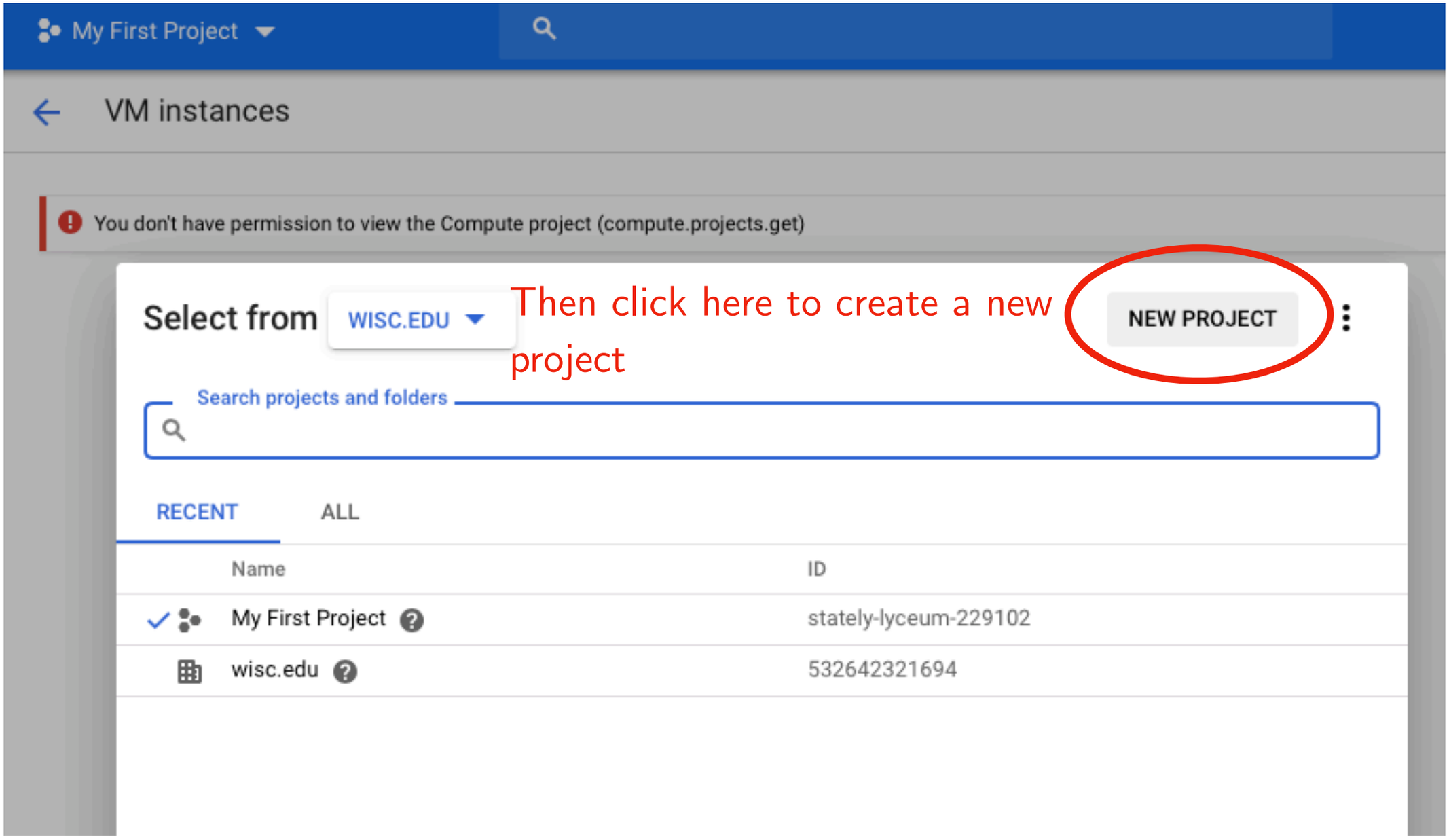


Option 2: Google Cloud Instances

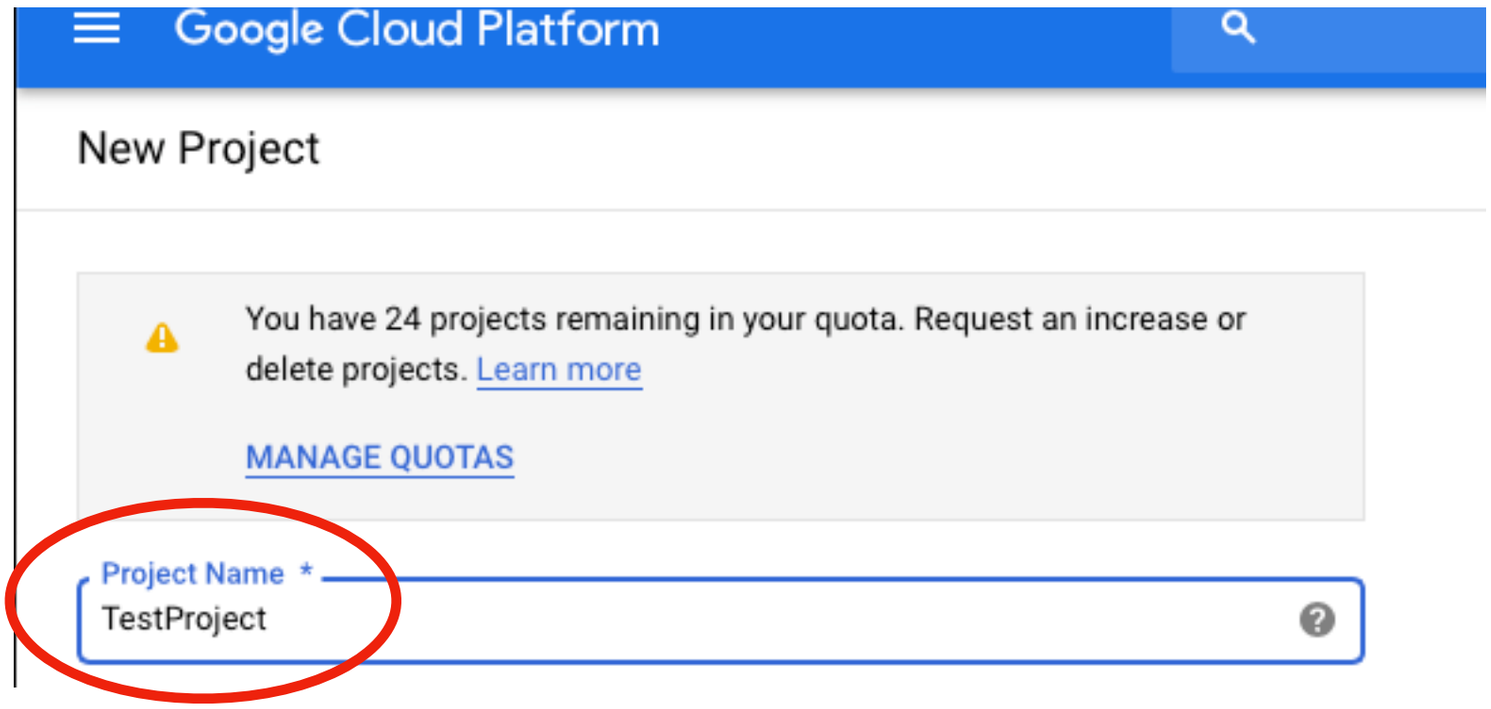
Click here to create a new project



Option 2: Google Cloud Instances

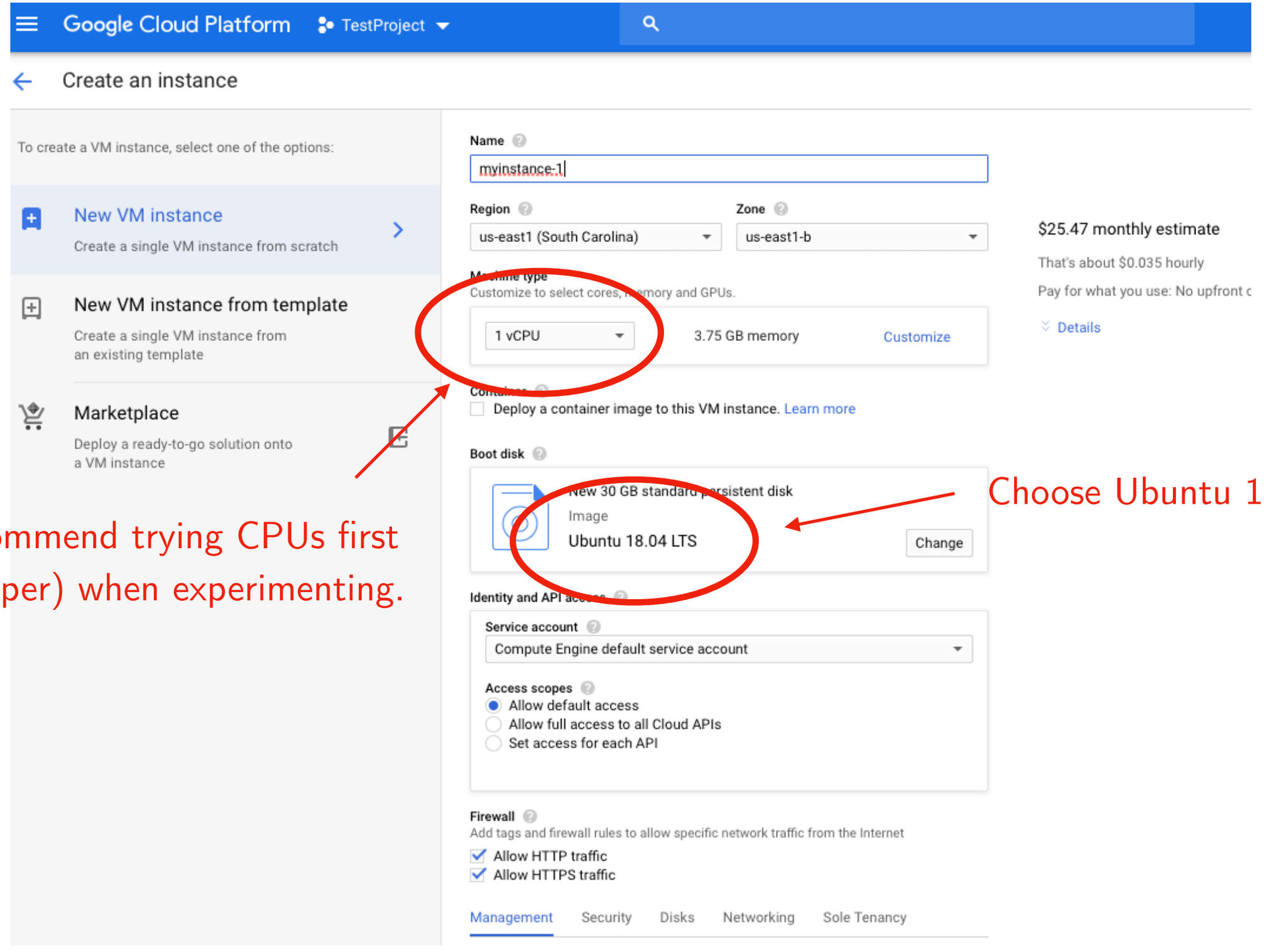


Option 2: Google Cloud Instances



Enter a name for your project

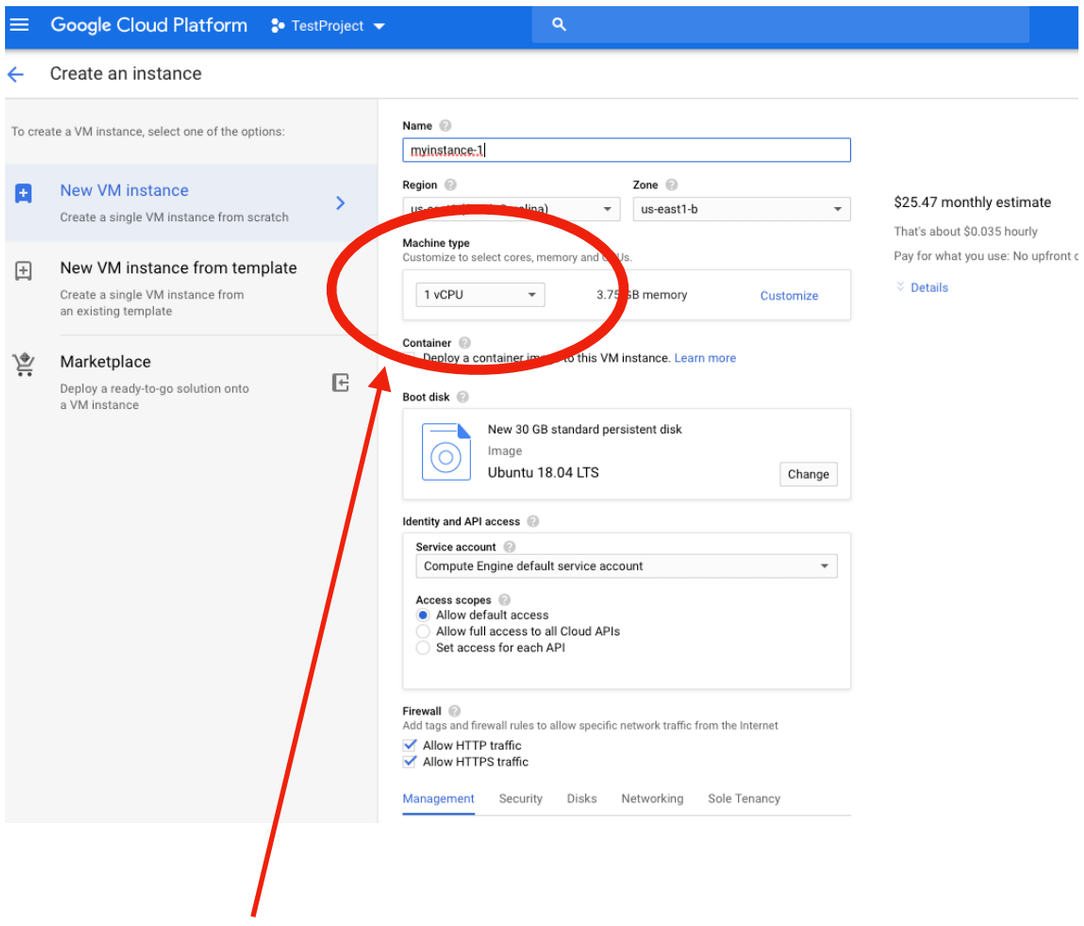
Option 2: Google Cloud Instances



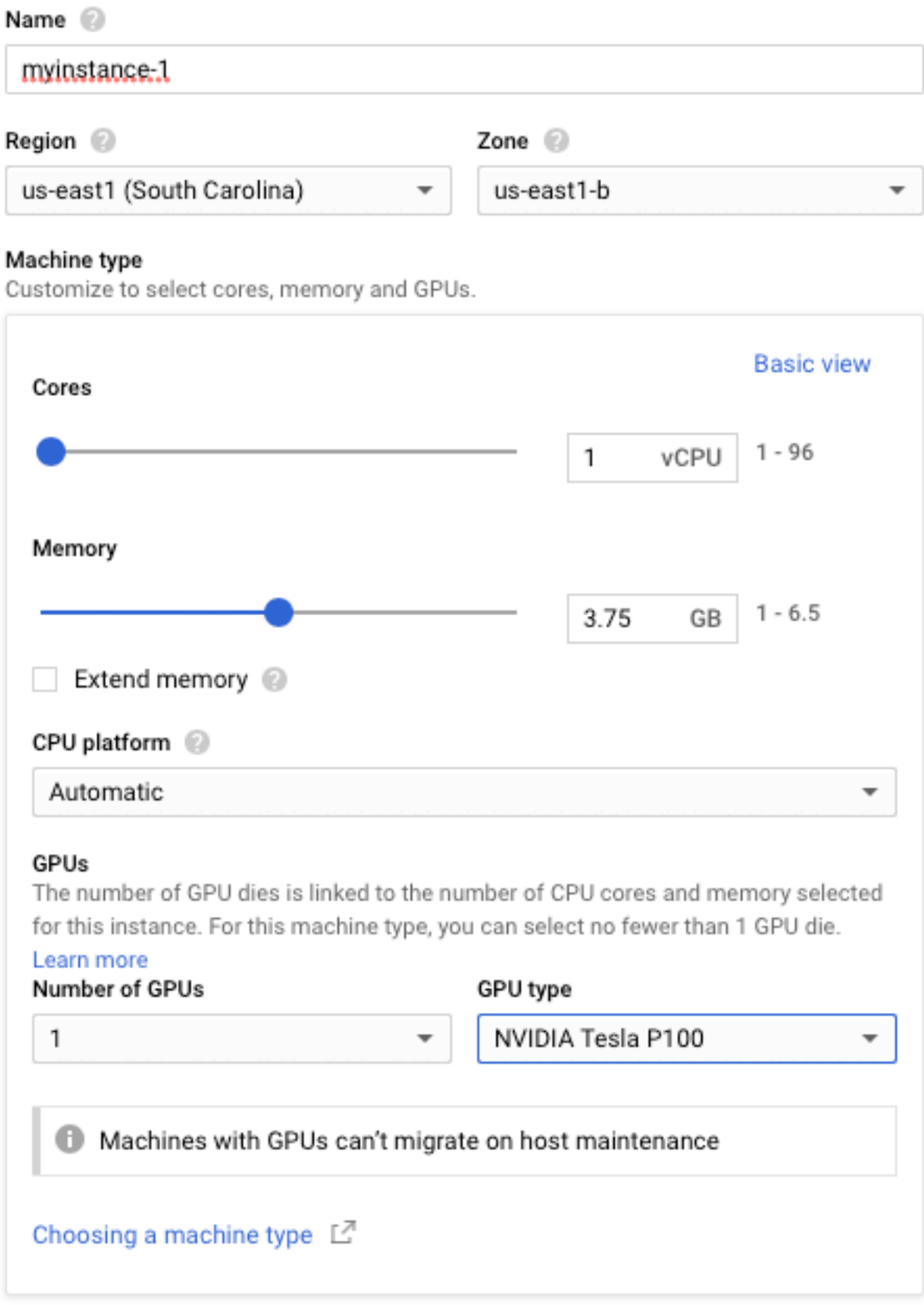
I recommend trying CPUs first (cheaper) when experimenting.

Choose Ubuntu 18.04 LTS

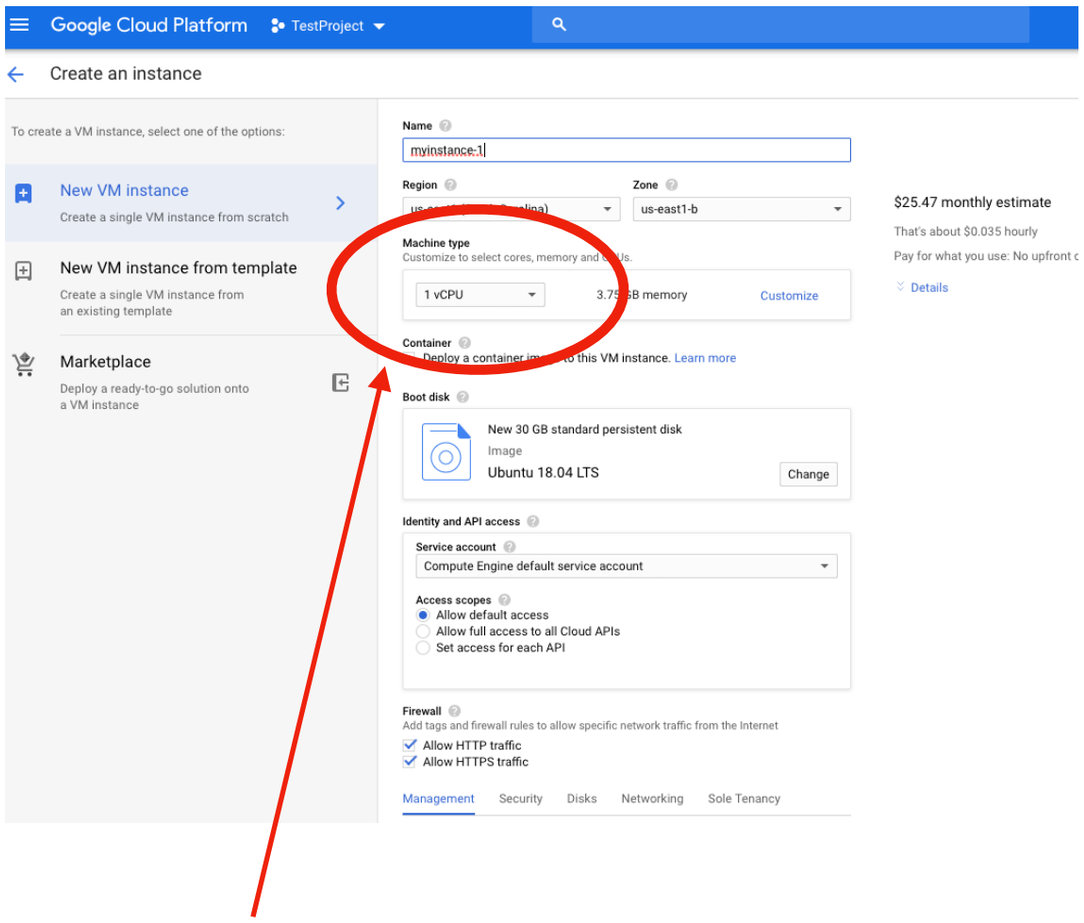
Option 2: Google Cloud Instances



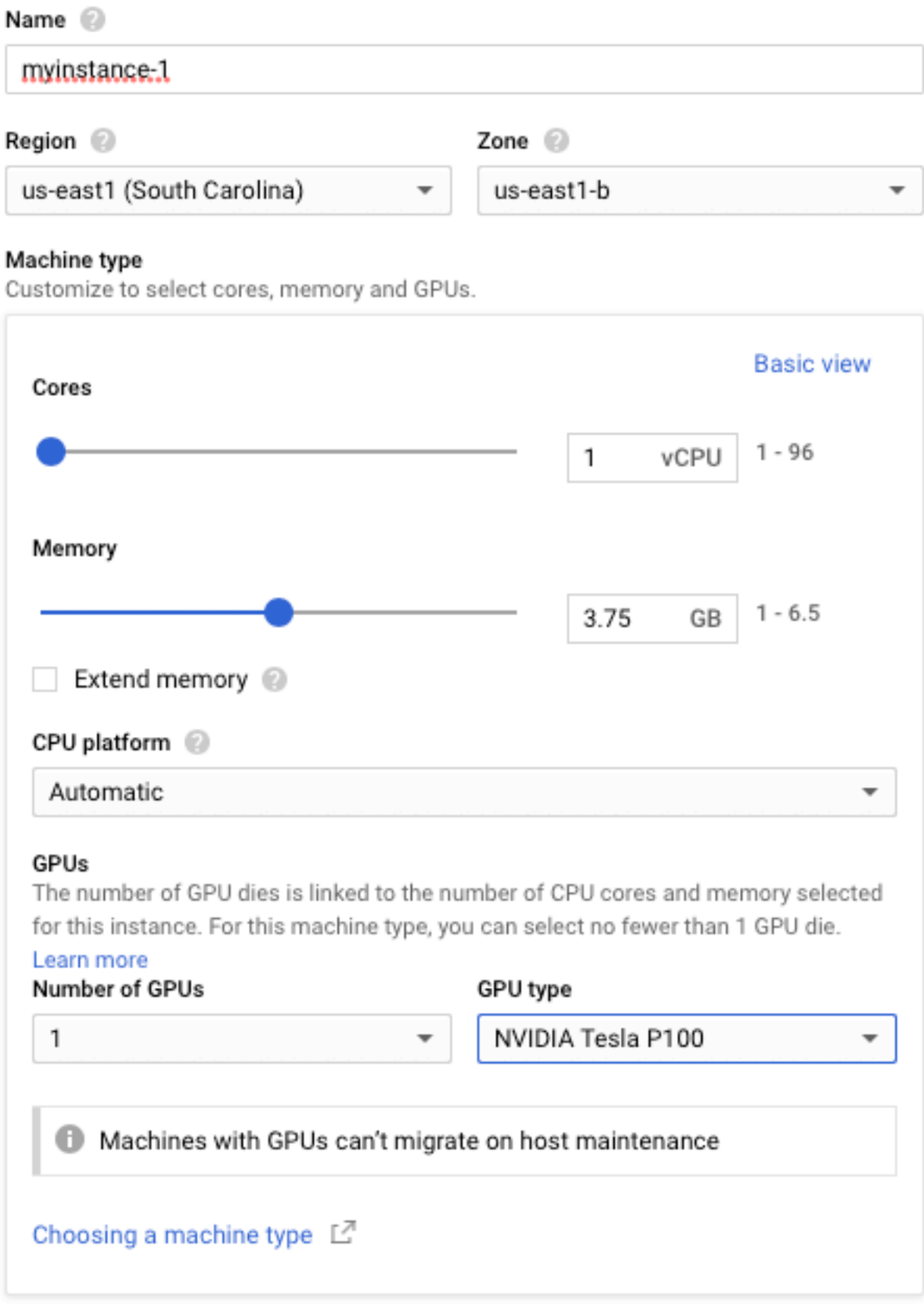
Later, when you need it in the class, you can choose GPUs from here



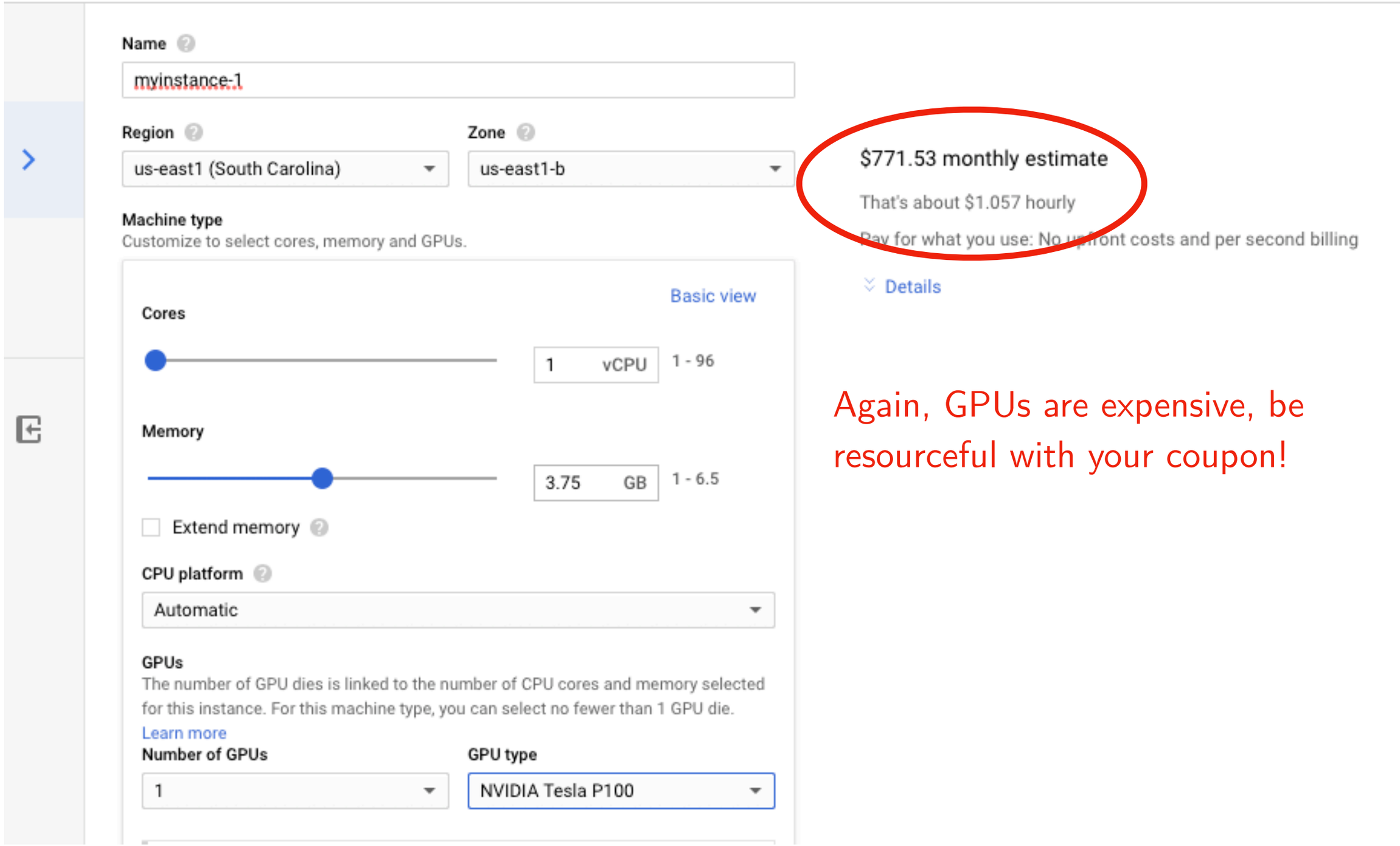
Option 2: Google Cloud Instances



Later, when you need it in the class, you can choose GPUs from here



Option 2: Google Cloud Instances

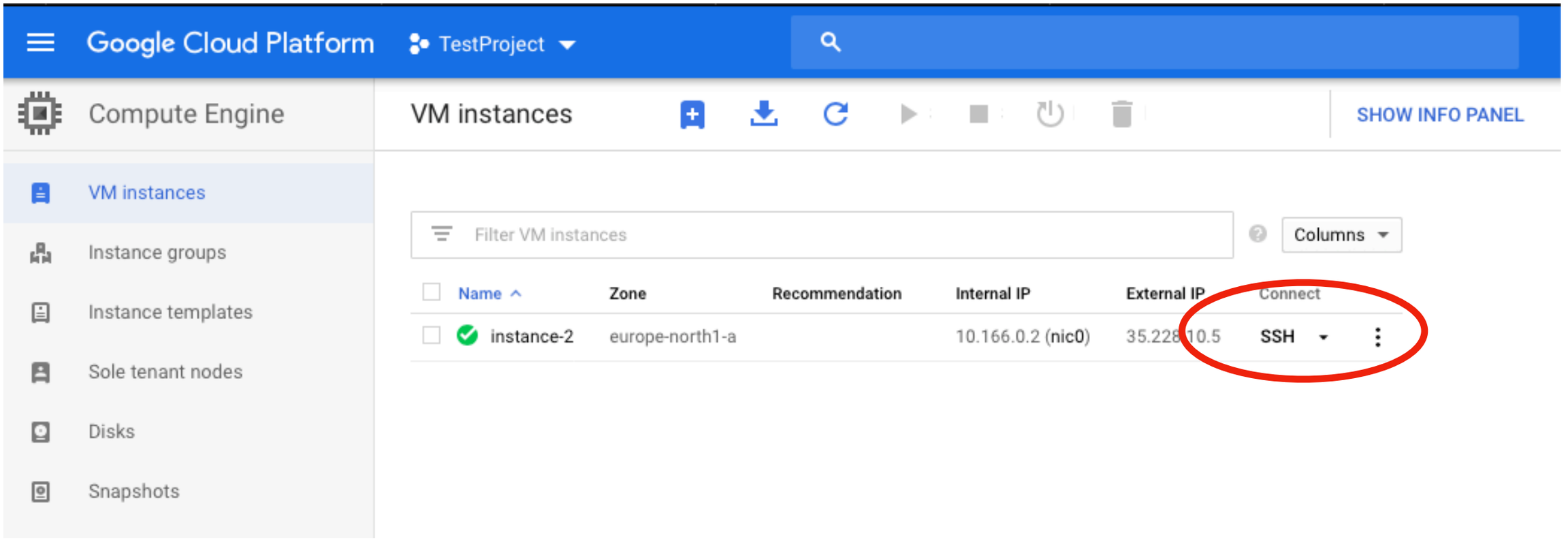


Again, GPUs are expensive, be resourceful with your coupon!

Option 2: Google Cloud Instances

It may unfortunately happen that sometimes all GPUs are busy (used by other people)

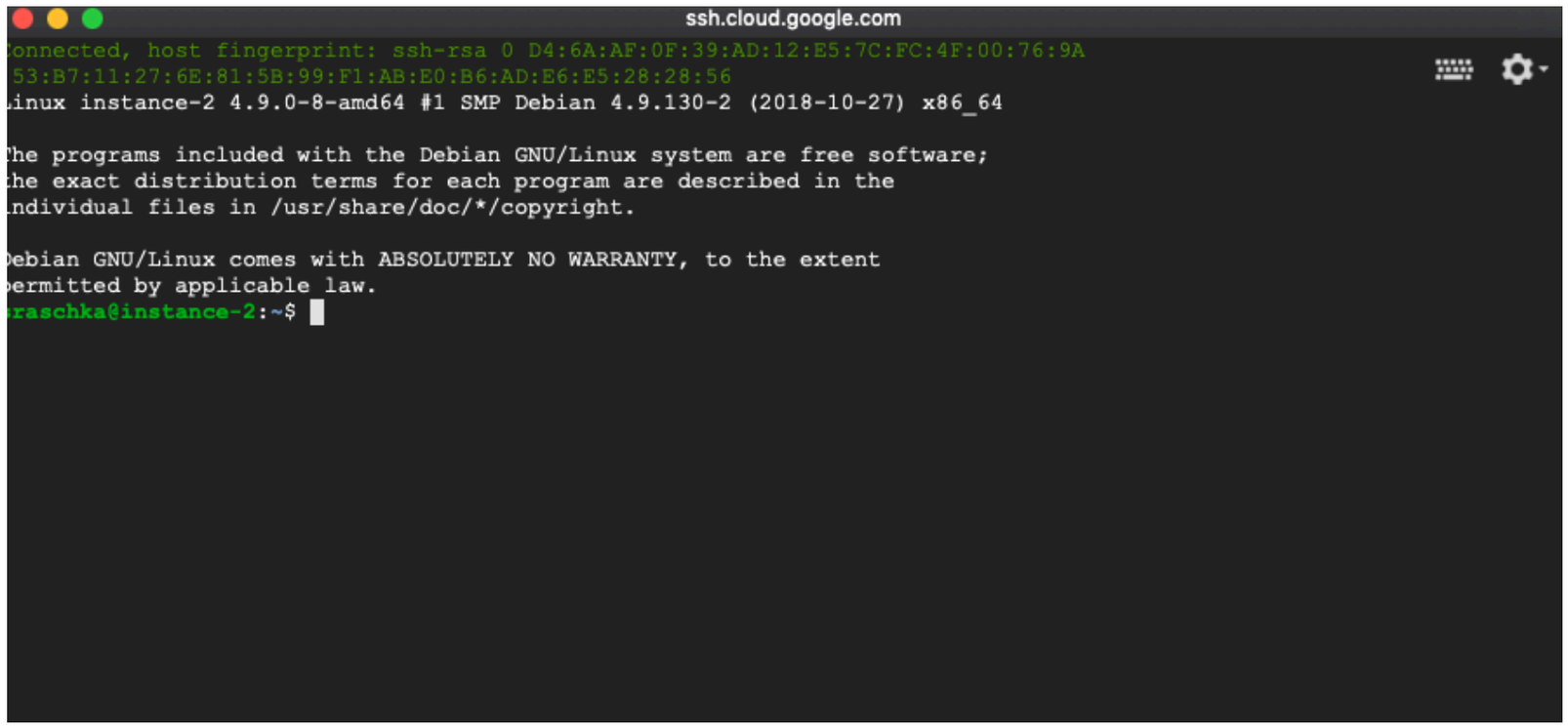
Option 2: Google Cloud Instances



Anyways, once your instance runs, you can click in SSH to log in

However, note that the first time you are planning to use GPUs on Google Cloud Engine, you need to submit a Quota request. You can it at
https://console.cloud.google.com/iam-admin/quotas

Option 2: Google Cloud Instances



It will basically be a Linux terminal

Option 2: Google Cloud Instances

First, I recommend installing conda. In this step, we download it first

Option 2: Google Cloud Instances

There is some package missing that we need for installing conda

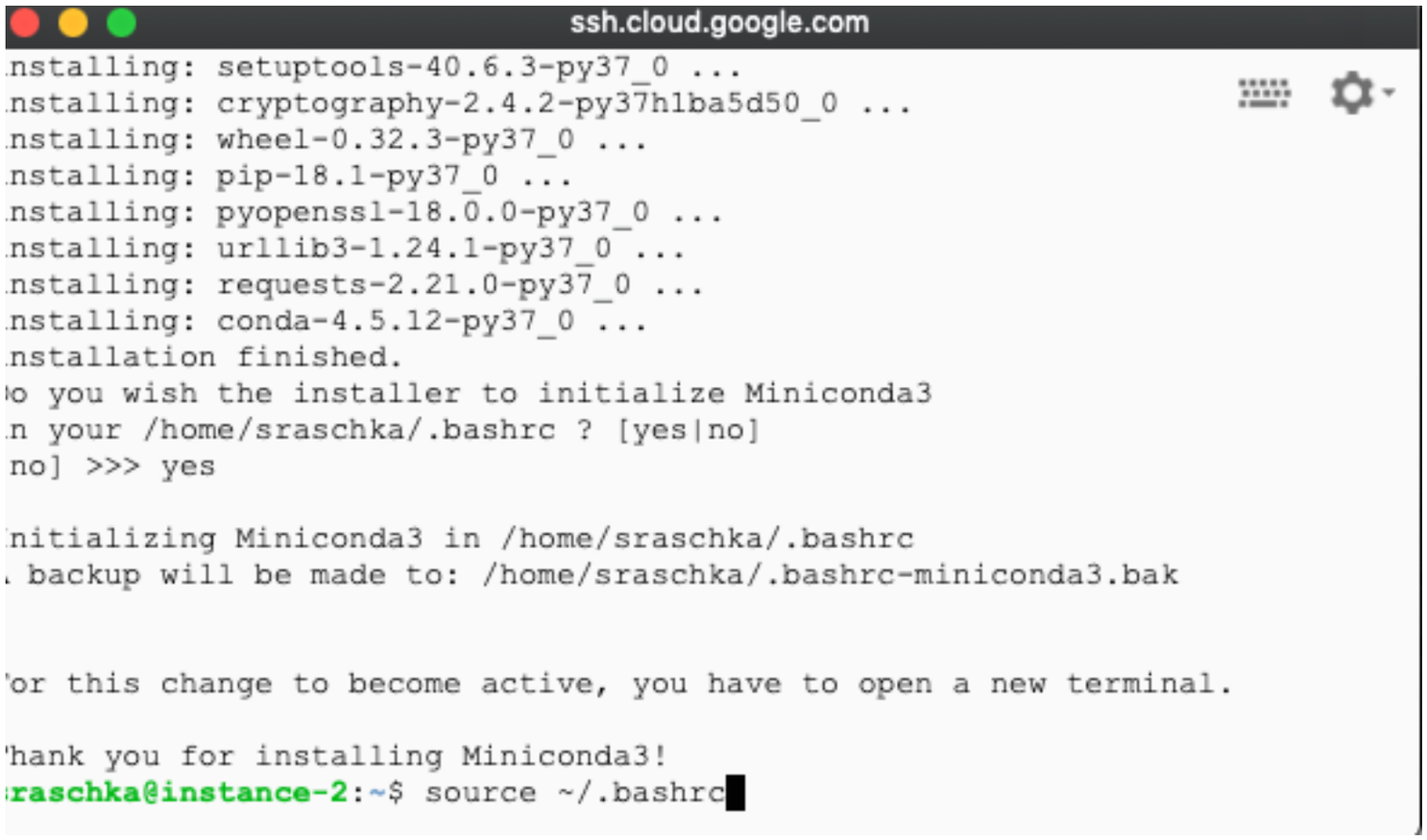
```
sraschka@instance-2:~$ sudo apt-get install bzip2
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  bzip2-doc
The following NEW packages will be installed:
  bzip2
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 47.5 kB of archives.
After this operation, 188 kB of additional disk space will be used.
Get:1 http://deb.debian.org/debian stretch/main amd64 bzip2 amd64 1.0.6-8.1 [47.5 kB]
Fetched 47.5 kB in 0s (761 kB/s)
Selecting previously unselected package bzip2.
(Reading database ... 34432 files and directories currently installed.)
Preparing to unpack .../bzip2_1.0.6-8.1_amd64.deb ...
Unpacking bzip2 (1.0.6-8.1) ...
Setting up bzip2 (1.0.6-8.1) ...
Processing triggers for man-db (2.7.6.1-2) ...
```

Option 2: Google Cloud Instances

After bzip2 is installed, you can run the installer for Miniconda:

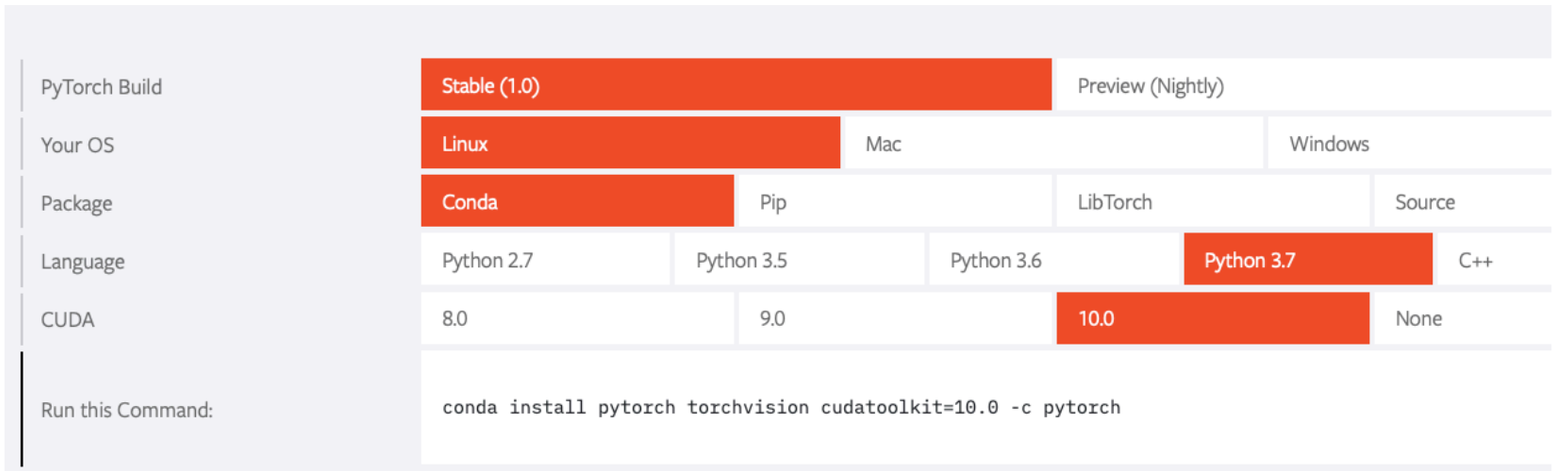
Option 2: Google Cloud Instances

After completing the installation, source your ~/.bashrc file



Option 2: Google Cloud Instances

Next, you can conveniently install PyTorch via the command from PyTorch's main website https://pytorch.org



Option 2: Google Cloud Instances

Next, let's check that PyTorch works
(you also may want to install ipython via conda):

```
xecuting transaction: done
(base) sraschka@instance-2:~$ ipython
Python 3.7.1 (default, Dec 14 2018, 19:28:38)
Type 'copyright', 'credits' or 'license' for more information
IPython 7.3.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: import torch

In [2]:
```

Note that GCE now also provides a tutorial and utilities for a more convenient use of PyTorch on their cloud instances. I recommend reading through the tutorial at https://cloud.google.com/deep-learning-vm/docs/pytorch_start_instance

Option 2: Google Cloud Instances

How can we get data onto that instance now? This is a bit tricky, you would maybe need some understanding of Linux or macOS's Unix. There are some tips here:

https://cloud.google.com/compute/docs/instances/connecting-to-instance

And here:

https://cloud.google.com/compute/docs/instances/transfer-files

Option 2: Google Cloud Instances

There, you need to follow the instructions to create authentication files:

SSH (LINUX & MACOS) PUTTY (WINDOWS)

To connect to an instance using `ssh` :

1. Provide your public SSH key to an instance using one of the available options.
2. In the console, find the external IP address for the instance that you want to connect to. Go to the lis

   GO TO THE INSTANCES PAGE

3. In a terminal, use the `ssh` command and your private SSH key file to connect to your instance. Spe external IP address of the instance that you want to connect to.

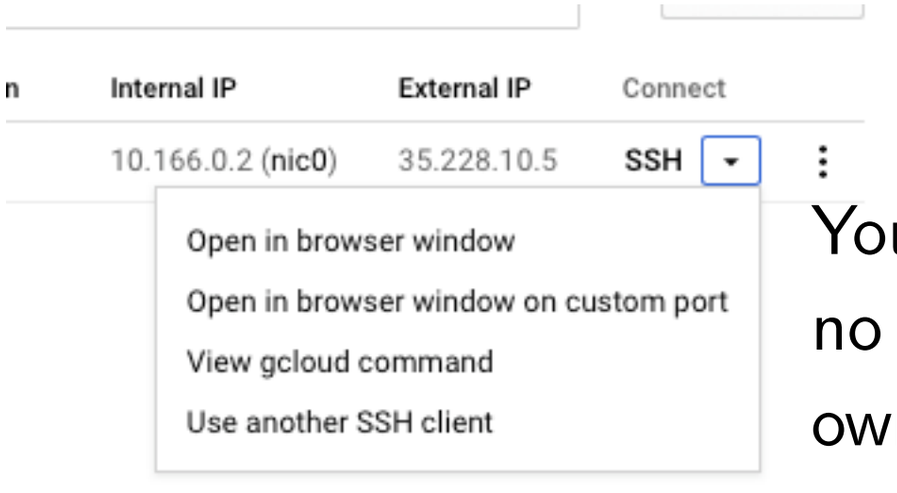
```
ssh -i [PATH_TO_PRIVATE_KEY] [USERNAME]@[EXTERNAL_IP_ADDRESS]
```

where:

- `[PATH_TO_PRIVATE_KEY]` is the path to your private SSH key file.
- `[USERNAME]` is the name of the user connecting to the instance. The username for your pub when the SSH key was created. You can connect to the instance as that user if the instance h that user and if you have the matching private SSH key.
- `[EXTERNAL_IP_ADDRESS]` is the external IP address for your instance.

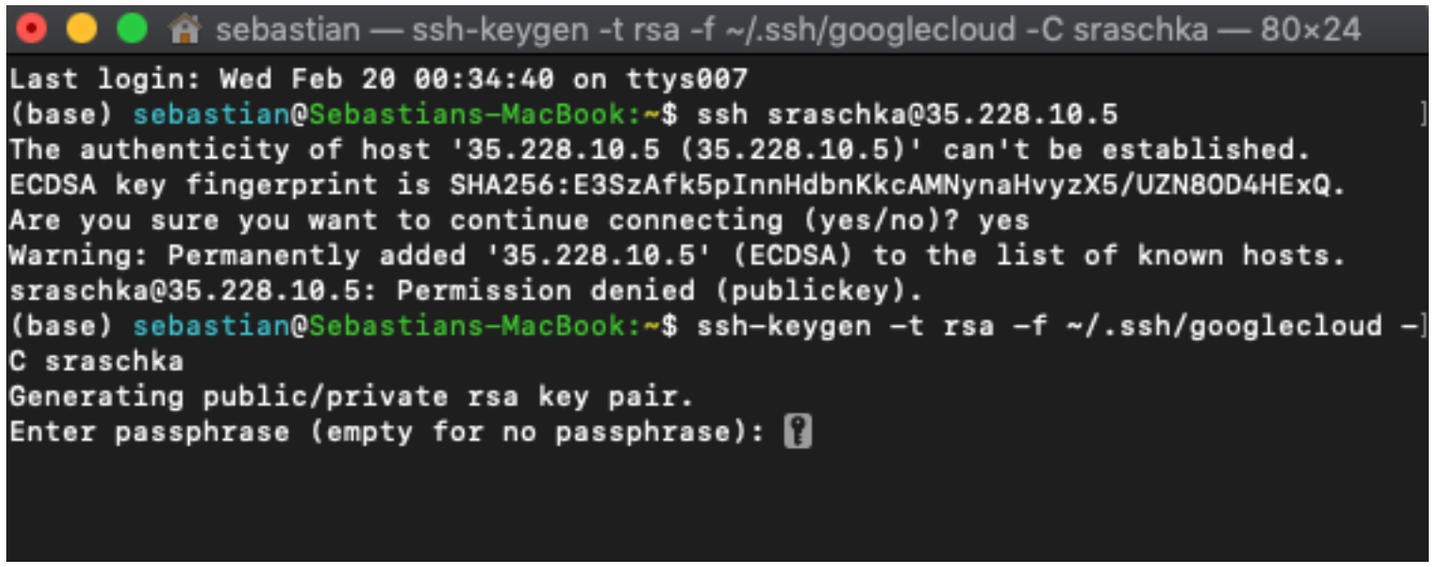
If the connection is successful, you can use the terminal to run commands on your instance. When command to disconnect from the instance.

Option 2: Google Cloud Instances



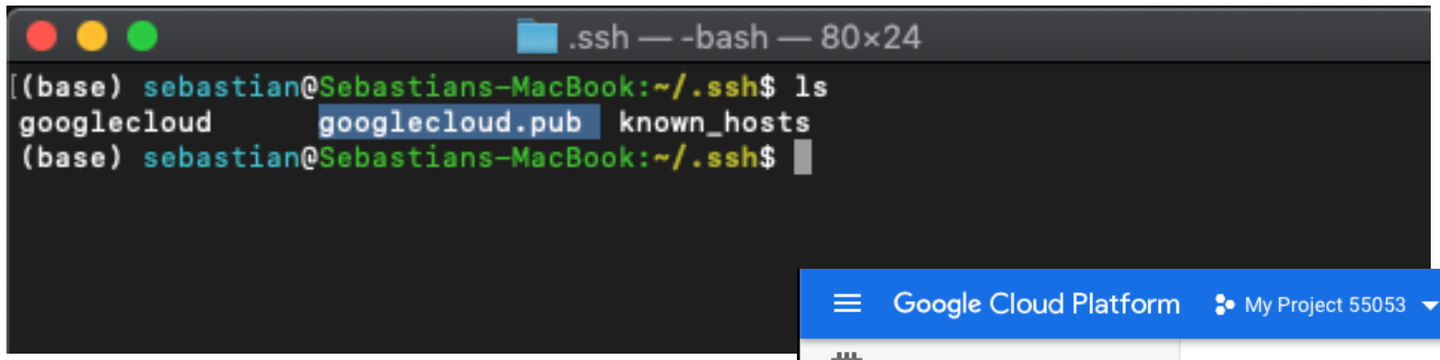
You can see that without key files, there's no access from your own terminal:

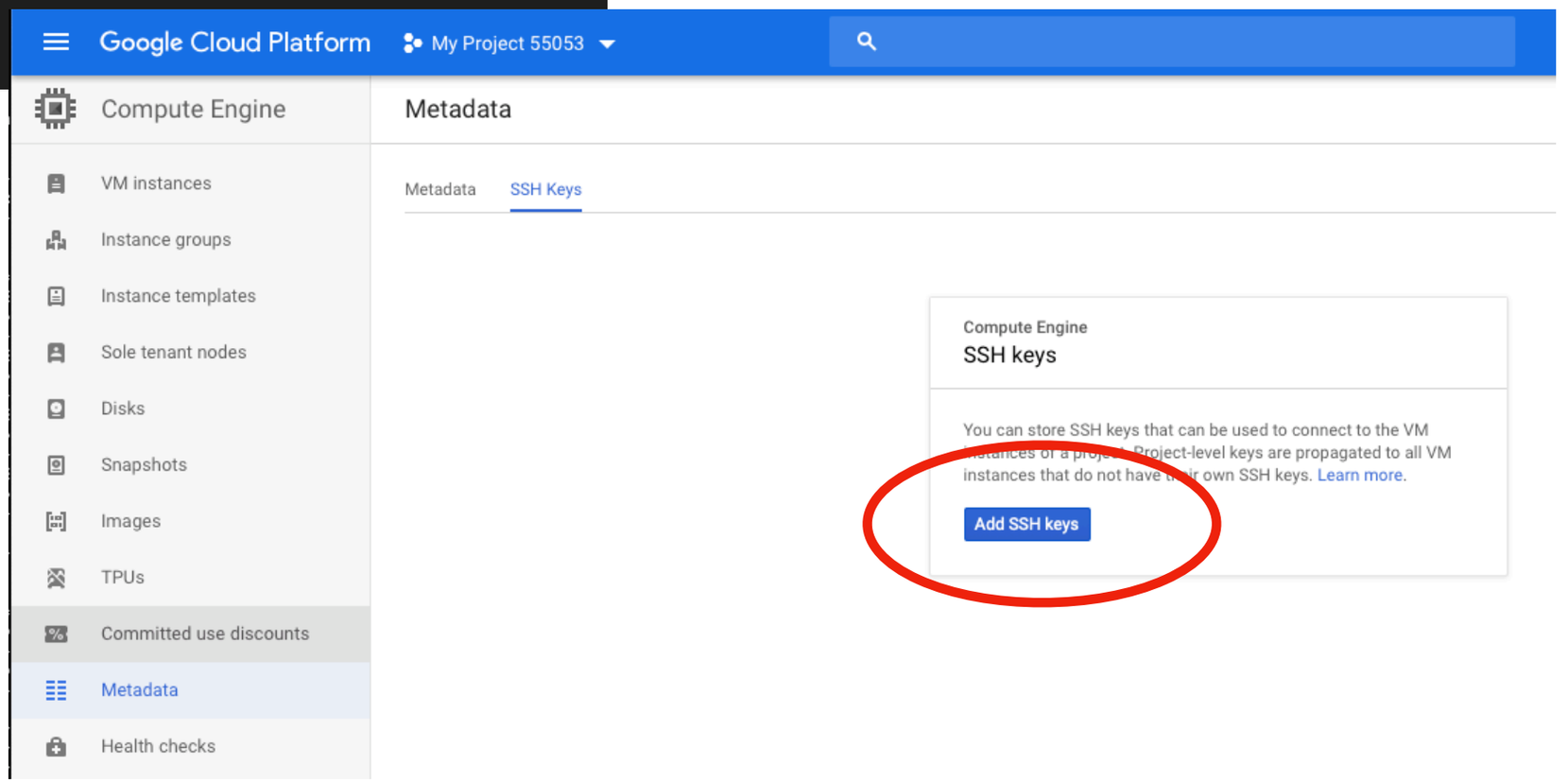
So, let's create a key pair:



Option 2: Google Cloud Instances

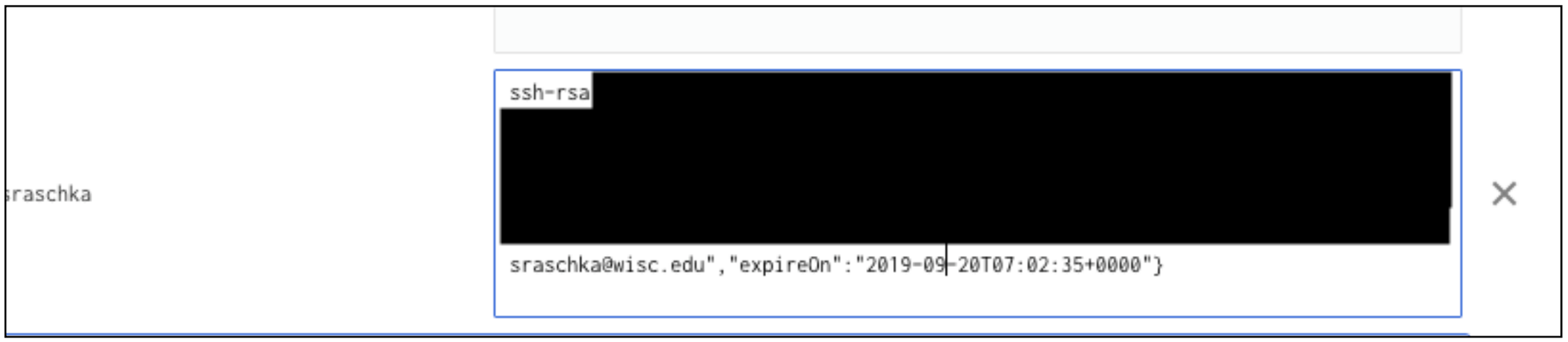
The public key (it's contents) is what needs to be entered online in your account:





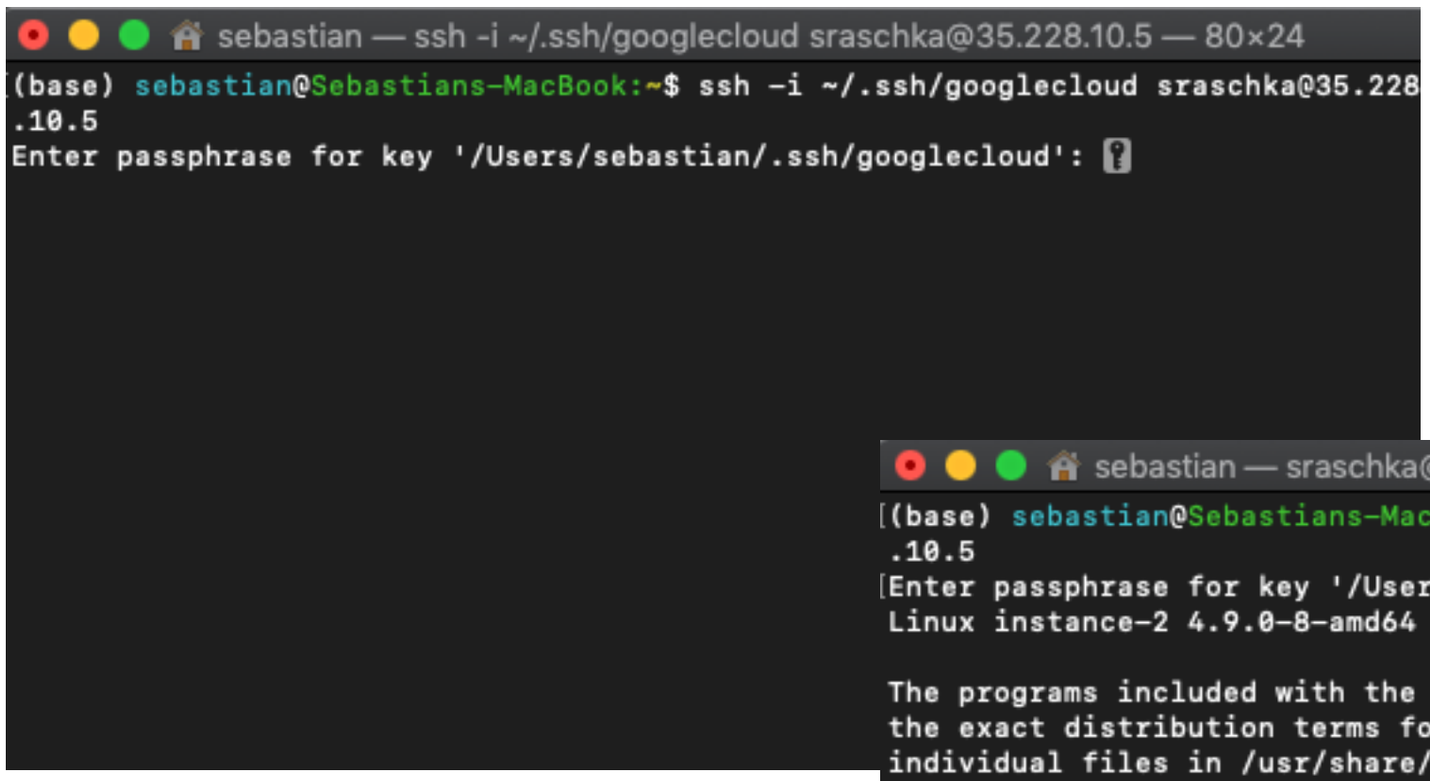
Option 2: Google Cloud Instances

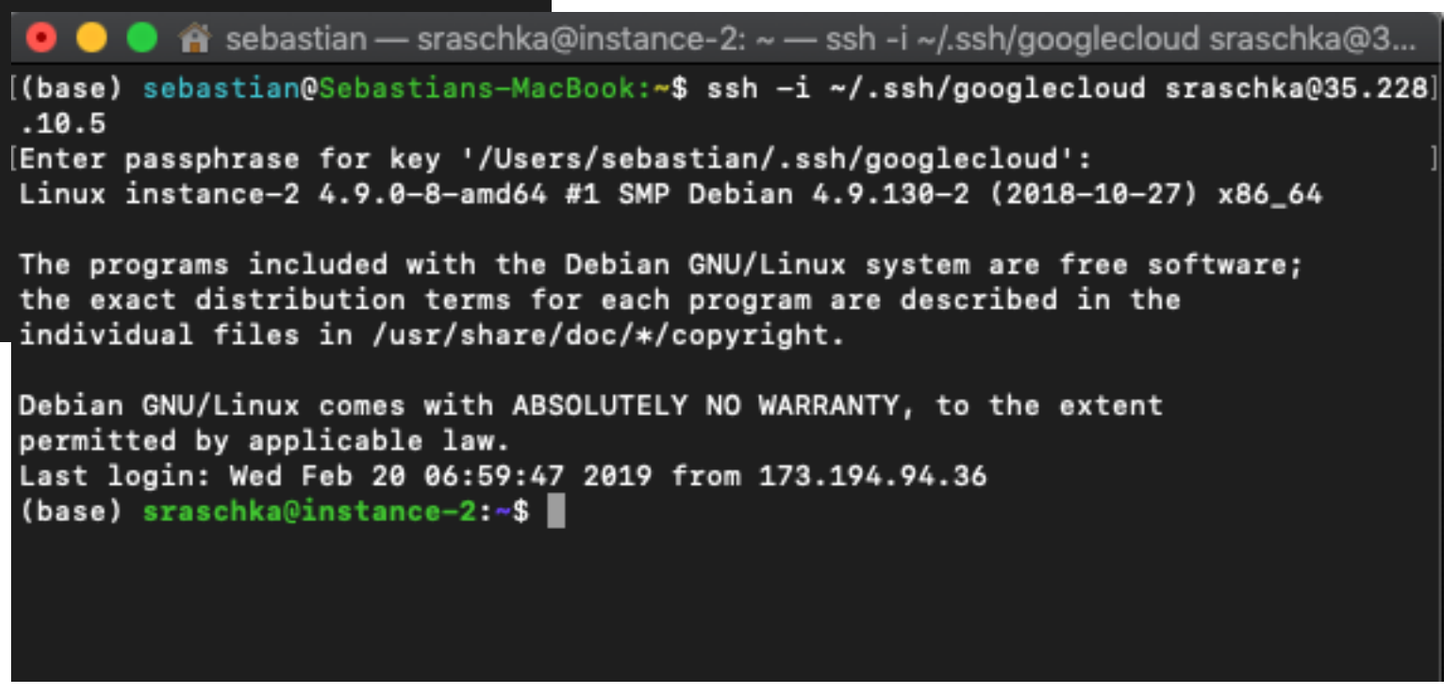
Note that I blacked out my key for security reasons ;)
Be aware of Google's special key formatting requirements (i.e., the contents you see in the curly braces; you may have to add that manually)



Option 2: Google Cloud Instances

Finally, we should be able to log in:



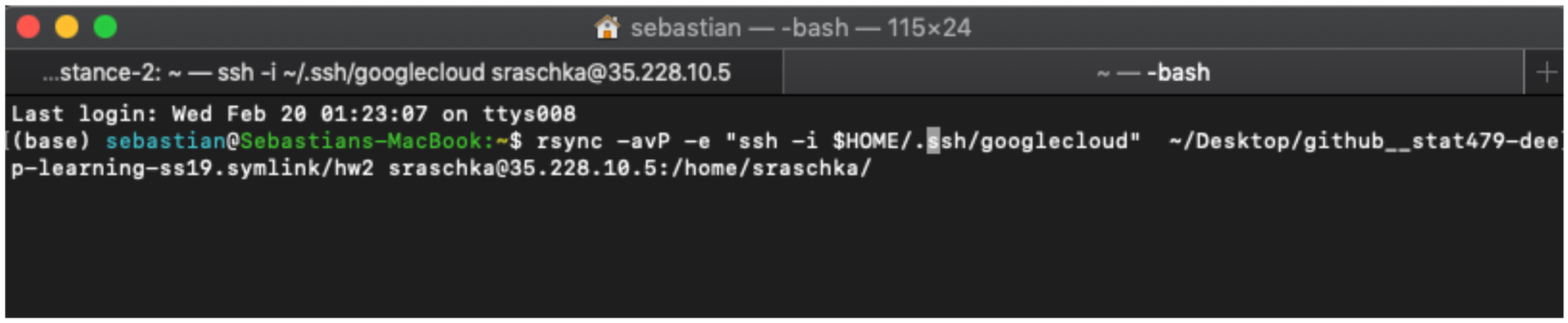


Option 2: Google Cloud Instances

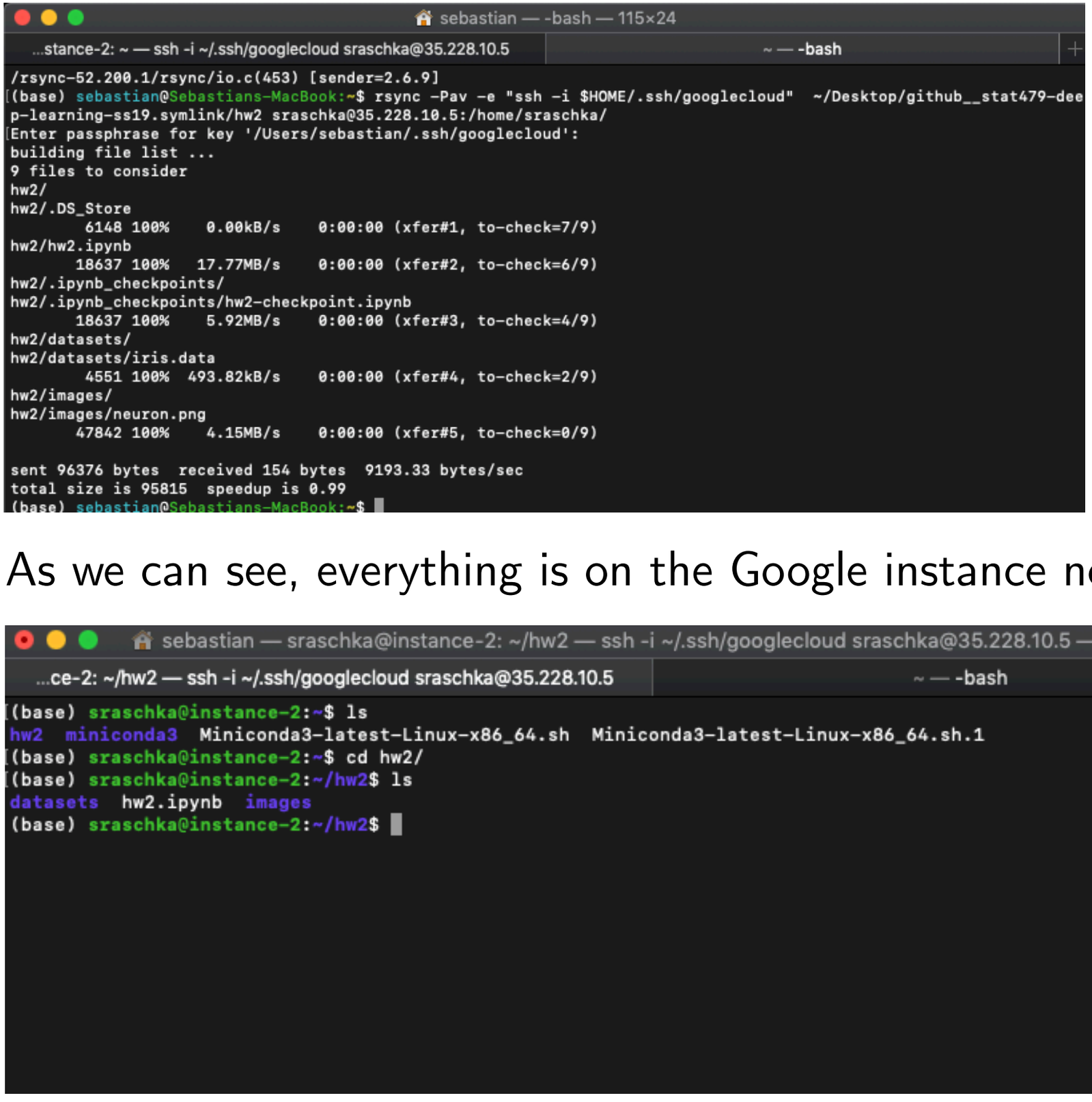
Next, you need to install rsync on the Google instance:

```
(base) sraschka@instance-2:~$ pwd
/home/sraschka
(base) sraschka@instance-2:~$ sudo apt-get install rsync
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  rsync
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 393 kB of archives.
After this operation, 703 kB of additional disk space will be used.
Get:1 http://deb.debian.org/debian stretch/main amd64 rsync amd64 3.1.2-1+deb9u1 [393 kB]
Fetched 393 kB in 0s (2,976 kB/s)
Selecting previously unselected package rsync.
(Reading database ... 34460 files and directories currently installed.)
Preparing to unpack .../rsync_3.1.2-1+deb9u1_amd64.deb ...
Unpacking rsync (3.1.2-1+deb9u1) ...
Setting up rsync (3.1.2-1+deb9u1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/rsync.service → /lib/systemd/system/rsync.service.
Processing triggers for systemd (232-25+deb9u8) ...
Processing triggers for man-db (2.7.6.1-2) ...
(base) sraschka@instance-2:~$
```

And after that, I can transfer files from my local machine to the Google instance:



Option 2: Google Cloud Instances



As we can see, everything is on the Google instance now:

Option 2: Google Cloud Instances

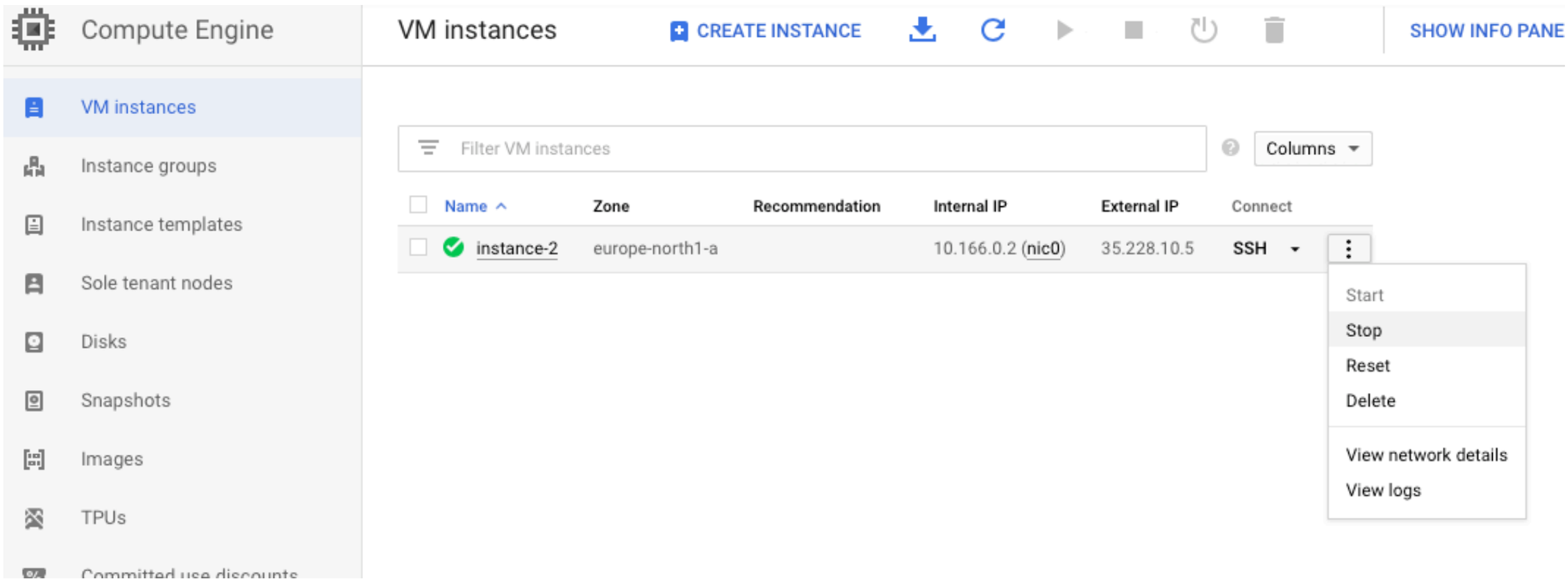
Setting up Jupyter Notebook access is also tricky. I uploaded some instructions here, which also apply to Google Cloud:

Section H.10 (pp. 25-27)

https://github.com/rasbt/stat479-deep-learning-ss19/blob/master/other/appendix_cloud-computing.pdf

Option 2: Google Cloud Instances

Very Important: When you are done, stop or delete your instances!



Once you are done, either stop or delete the instances. Stopped instances will cost some minor amount for storage, but you won't have to redo all the steps. Deleted instances are gone forever. I recommend stopping the instance until the end of the class if you like to reuse it.